1091.ch08 5/13/03 1:22 PM Page 413

8 Network Models

Many important optimization problems can best be analyzed by means of a graphical or network representation. In this chapter, we consider four specific network models—shortest-path problems, maximum-flow problems, CPM–PERT project-scheduling models, and minimum-spanning tree problems—for which efficient solution procedures exist. We also discuss minimum-cost network flow problems (MCNFPs), of which transportation, assignment, transshipment, shortest-path, and maximum-flow problems and the CPM project-scheduling models are all special cases. Finally, we discuss a generalization of the transportation simplex, the network simplex, which can be used to solve MCNFPs. We begin the chapter with some basic terms used to describe graphs and networks.

8.1 Basic Definitions

A **graph,** or **network,** is defined by two sets of symbols: nodes and arcs. First, we define a set (call it V) of points, or **vertices.** The vertices of a graph or network are also called **nodes.**

We also define a set of arcs A.

DEFINITION ■ An **arc** consists of an ordered pair of vertices and represents a possible direction of motion that may occur between vertices. ■

For our purposes, if a network contains an arc (j, k), then motion is possible from node j to node k. Suppose nodes 1, 2, 3, and 4 of Figure 1 represent cities, and each arc represents a (one-way) road linking two cities. For this network, $V = \{1, 2, 3, 4\}$ and $A = \{(1, 2), (2, 3), (3, 4), (4, 3), (4, 1)\}$. For the arc (j, k), node j is the **initial node,** and node k is the **terminal node.** The arc (j, k) is said to go from node j to node k. Thus, the arc (2, 3) has initial node 2 and terminal node 3, and it goes from node 2 to node 3. The arc (2, 3) may be thought of as a (one-way) road on which we may travel from city 2 to city 3. In Figure 1, the arcs show that travel is allowed from city 3 to city 4, and from city 4 to city 3, but that travel between the other cities may be one way only.

Later, we often discuss a group or collection of arcs. The following definitions are convenient ways to describe certain groups or collections of arcs.

DEFINITION ■ A sequence of arcs such that every arc has exactly one vertex in common with the previous arc is called a **chain.** ■

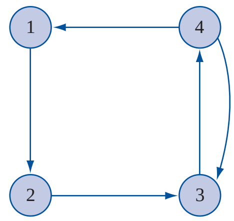


FIGURE 1
Example of a Network

DEFINITION ■ A **path** is a chain in which the terminal node of each arc is identical to the initial node of the next arc. ■

For example, in Figure 1, (1, 2)–(2, 3)–(4, 3) is a chain but not a path; (1, 2)–(2, 3)–(3, 4) is a chain *and* a path. The path (1, 2)–(2, 3)–(3, 4) represents a way to travel from node 1 to node 4.

8.2 Shortest-Path Problems

In this section, we assume that each arc in the network has a length associated with it. Suppose we start at a particular node (say, node 1). The problem of finding the shortest path (path of minimum length) from node 1 to any other node in the network is called a **shortest-path problem.** Examples 1 and 2 are shortest-path problems.

EXAMPLE 1 Shortest Path

Let us consider the Powerco example (Figure 2). Suppose that when power is sent from plant 1 (node 1) to city 1 (node 6), it must pass through relay substations (nodes 2–5). For any pair of nodes between which power can be transported, Figure 2 gives the distance (in miles) between the nodes. Thus, substations 2 and 4 are 3 miles apart, and power cannot be sent between substations 4 and 5. Powerco wants the power sent from plant 1 to city 1 to travel the minimum possible distance, so it must find the shortest path in Figure 2 that joins node 1 to node 6.

If the cost of shipping power were proportional to the distance the power travels, then knowing the shortest path between plant 1 and city 1 in Figure 2 (and the shortest path between plant i and city j in similar diagrams) would be necessary to determine the shipping costs for the transportation version of the Powerco problem discussed in Chapter 7.

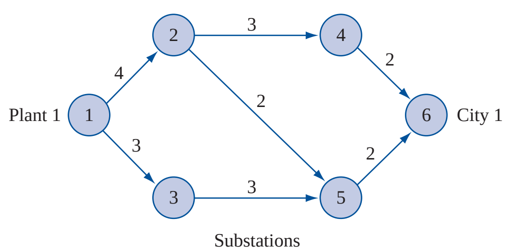


FIGURE 2
Network for Powerco

EXAMPLE 2 Equipment Replacement

I have just purchased (at time 0) a new car for $12,000. The cost of maintaining a car during a year depends on its age at the beginning of the year, as given in Table 1. To avoid the high maintenance costs associated with an older car, I may trade in my car and purchase a new car. The price I receive on a trade-in depends on the age of the car at the time of trade-in (see Table 2). To simplify the computations, we assume that at any time, it costs $12,000 to purchase a new car. My goal is to minimize the net cost (purchasing costs + maintenance costs − money received in trade-ins) incurred during the next five years. Formulate this problem as a shortest-path problem.

Solution Our network will have six nodes (1, 2, 3, 4, 5, and 6). Node i is the beginning of year i. For $i < j$, an arc (i, j) corresponds to purchasing a new car at the beginning of year i and keeping it until the beginning of year j. The length of arc (i, j) (call it c_{ij}) is the total net cost incurred in owning and operating a car from the beginning of year i to the beginning of year j if a new car is purchased at the beginning of year i and this car is traded in for a new car at the beginning of year j. Thus,

$$\begin{aligned} c_{ij} = {} & \text{maintenance cost incurred during years } i, i + 1, \ldots, j - 1 \\ & + \text{cost of purchasing car at beginning of year } i \\ & - \text{trade-in value received at beginning of year } j \end{aligned}$$

Applying this formula to the information in the problem yields (all costs are in thousands)

$$c_{12} = 2 + 12 - 7 = 7 \qquad c_{16} = 2 + 4 + 5 + 9 + 12 + 12 - 0 = 44$$
$$c_{13} = 2 + 4 + 12 - 6 = 12 \qquad c_{23} = 2 + 12 - 7 = 7$$
$$c_{14} = 2 + 4 + 5 + 12 - 2 = 21 \qquad c_{24} = 2 + 4 + 12 - 6 = 12$$
$$c_{15} = 2 + 4 + 5 + 9 + 12 - 1 = 31 \qquad c_{25} = 2 + 4 + 5 + 12 - 2 = 21$$

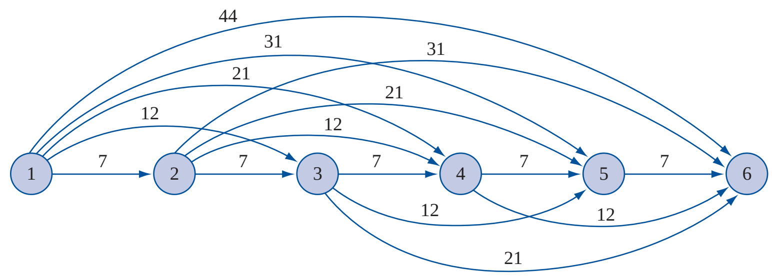
TABLE 1
Car Maintenance Costs

Age of Car (Years)	Annual Maintenance Cost ($)
0	2,000
1	4,000
2	5,000
3	9,000
4	12,000

TABLE 2
Car Trade-in Prices

Age of Car (Years)	Trade-in Price
1	7,000
2	6,000
3	2,000
4	1,000
5	0

FIGURE 3
Network for Minimizing Car Costs



$c_{26} = 2 + 4 + 5 + 9 + 12 - 1 = 31$ $\quad$ $c_{45} = 2 + 12 - 7 = 7$

$c_{34} = 2 + 12 - 7 = 7$ $\quad$ $c_{46} = 2 + 4 + 12 - 6 = 12$

$c_{35} = 2 + 4 + 12 - 6 = 12$ $\quad$ $c_{56} = 2 + 12 - 7 = 7$

$c_{36} = 2 + 4 + 5 + 12 - 2 = 21$

We now see that the length of any path from node 1 to node 6 is the net cost incurred during the next five years corresponding to a particular trade-in strategy. For example, suppose I trade in the car at the beginning of year 3 and next trade in the car at the end of year 5 (the beginning of year 6). This strategy corresponds to the path 1–3–6 in Figure 3. The length of this path ($c_{13} + c_{36}$) is the total net cost incurred during the next five years if I trade in the car at the beginning of year 3 and at the beginning of year 6. Thus, the length of the shortest path from node 1 to node 6 in Figure 3 is the minimum net cost that can be incurred in operating a car during the next five years.

Dijkstra's Algorithm

Assuming that all arc lengths are nonnegative, the following method, known as **Dijkstra's algorithm,** can be used to find the shortest path from a node (say, node 1) to all other nodes. To begin, we label node 1 with a permanent label of 0. Then we label each node i that is connected to node 1 by a single arc with a "temporary" label equal to the length of the arc joining node 1 to node i. Each other node (except, of course, for node 1) will have a temporary label of ∞. Choose the node with the smallest temporary label and make this label permanent.

Now suppose that node i has just become the $(k + 1)$th node to be given a permanent label. Then node i is the kth closest node to node 1. At this point, the temporary label of any node (say, node i') is the length of the shortest path from node 1 to node i' that passes only through nodes contained in the $k - 1$ closest nodes to node 1. For each node j that now has a temporary label and is connected to node i by an arc, we replace node j's temporary label with

$$\min \begin{cases} \text{node } j\text{'s current temporary label} \\ \text{node } i\text{'s permanent label} + \text{length of arc } (i, j) \end{cases}$$

(Here, $\min\{a, b\}$ is the smaller of a and b.) The new temporary label for node j is the length of the shortest path from node 1 to node j that passes only through nodes contained in the k closest nodes to node 1. We now make the smallest temporary label a permanent label. The node with this new permanent label is the $(k + 1)$th closest node to node 1. Continue this process until all nodes have a permanent label. To find the shortest path from node 1 to node j, work backward from node j by finding nodes having labels differing by exactly the length of the connecting arc. Of course, if we want the shortest path from node 1 to node j, we can stop the labeling process as soon as node j receives a permanent label.

To illustrate Dijkstra's algorithm, we find the shortest path from node 1 to node 6 in Figure 2. We begin with the following labels (a * represents a permanent label, and the ith number is the label of the node i): $[0^* \quad 4 \quad 3 \quad \infty \quad \infty \quad \infty]$. Node 3 now has the smallest temporary label. We therefore make node 3's label permanent and obtain the following labels:

$$[0^* \quad 4 \quad 3^* \quad \infty \quad \infty \quad \infty]$$

We now know that node 3 is the closest node to node 1. We compute new temporary labels for all nodes that are connected to node 3 by a single arc. In Figure 2 that is node 5.

$$\text{New node 5 temporary label} = \min\{\infty, 3 + 3\} = 6$$

Node 2 now has the smallest temporary label; we now make node 2's label permanent. We now know that node 2 is the second closest node to node 1. Our new set of labels is

$$[0^* \quad 4^* \quad 3^* \quad \infty \quad 6 \quad \infty]$$

Because nodes 4 and 5 are connected to the newly permanently labeled node 2, we must change the temporary labels of nodes 4 and 5. Node 4's new temporary label is $\min\{\infty, 4 + 3\} = 7$ and node 5's new temporary label is $\min\{6, 4 + 2\} = 6$. Node 5 now has the smallest temporary label, so we make node 5's label permanent. We now know that node 5 is the third closest node to node 1. Our new labels are

$$[0^* \quad 4^* \quad 3^* \quad 7 \quad 6^* \quad \infty]$$

Only node 6 is connected to node 5, so node 6's temporary label will change to $\min\{\infty, 6 + 2\} = 8$. Node 4 now has the smallest temporary label, so we make node 4's label permanent. We now know that node 4 is the fourth closest node to node 1. Our new labels are

$$[0^* \quad 4^* \quad 3^* \quad 7^* \quad 6^* \quad 8]$$

Because node 6 is connected to the newly permanently labeled node 4, we must change node 6's temporary label to $\min\{8, 7 + 2\} = 8$. We can now make node 6's label permanent. Our final set of labels is $[0^* \quad 4^* \quad 3^* \quad 7^* \quad 6^* \quad 8^*]$. We can now work backward and find the shortest path from node 1 to node 6. The difference between node 6's and node 5's permanent labels is 2 = length of arc (5, 6), so we go back to node 5. The difference between node 5's and node 2's permanent labels is 2 = length of arc (2, 5), so we may go back to node 2. Then, of course, we must go back to node 1. Thus, 1–2–5–6 is a shortest path (of length 8) from node 1 to node 6. Observe that when we were at node 5, we could also have worked backward to node 3 and obtained the shortest path 1–3–5–6.

The Shortest-Path Problem as a Transshipment Problem

Finding the shortest path between node i and node j in a network may be viewed as a transshipment problem. Simply try to minimize the cost of sending one unit from node i to node j (with all other nodes in the network being transshipment points), where the cost of sending one unit from node k to node k' is the length of arc (k, k') if such an arc exists and is M (a large positive number) if such an arc does not exist. As in Section 7.6, the cost of shipping one unit from a node to itself is zero. Following the method described in Section 7.6, this transshipment problem may be transformed into a balanced transportation problem.

TABLE 3
Transshipment Representation of Shortest-Path Problem and Optimal Solution (1)

Node	2	3	4	5	6	Supply
1	$\boxed{4}$ 1	$\boxed{3}$	$\boxed{M}$	$\boxed{M}$	$\boxed{M}$	1
2	$\boxed{0}$	$\boxed{M}$	$\boxed{3}$	$\boxed{2}$ 1	$\boxed{M}$	1
3	$\boxed{M}$	$\boxed{0}$ 1	$\boxed{M}$	$\boxed{3}$	$\boxed{M}$	1
4	$\boxed{M}$	$\boxed{M}$	$\boxed{0}$ 1	$\boxed{M}$	$\boxed{2}$	1
5	$\boxed{M}$	$\boxed{M}$	$\boxed{M}$	$\boxed{0}$	$\boxed{2}$ 1	1
Demand	1	1	1	1	1	

To illustrate the preceding ideas, we formulate the balanced transportation problem associated with finding the shortest path from node 1 to node 6 in Figure 2. We want to send one unit from node 1 to node 6. Node 1 is a supply point, node 6 is a demand point, and nodes 2, 3, 4, and 5 will be transshipment points. Using $s = 1$, we obtain the balanced transportation problem shown in Table 3. This transportation problem has two optimal solutions:

1 $z = 4 + 2 + 2 = 8$, $x_{12} = x_{25} = x_{56} = x_{33} = x_{44} = 1$ (all other variables equal 0). This solution corresponds to the path 1–2–5–6.

2 $z = 3 + 3 + 2 = 8$, $x_{13} = x_{35} = x_{56} = x_{22} = x_{44} = 1$ (all other variables equal 0). This solution corresponds to the path 1–3–5–6.

REMARK After formulating a shortest-path problem as a transshipment problem, the problem may be solved easily by using LINGO or a spreadsheet optimizer. See Section 7.1 for details.

PROBLEMS

Group A

1 Find the shortest path from node 1 to node 6 in Figure 3.

2 Find the shortest path from node 1 to node 5 in Figure 4.

3 Formulate Problem 2 as a transshipment problem.

4 Use Dijkstra's algorithm to find the shortest path from node 1 to node 4 in Figure 5. Why does Dijkstra's algorithm fail to obtain the correct answer?

FIGURE 4
Network for Problem 2

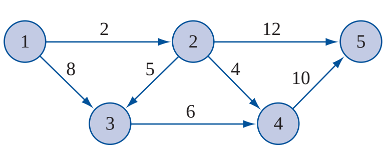
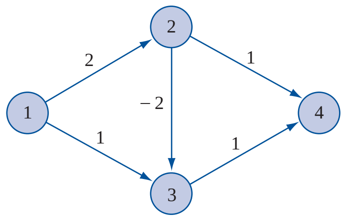


FIGURE 5
Network for Problem 4



5 Suppose it costs $10,000 to purchase a new car. The annual operating cost and resale value of a used car are shown in Table 4. Assuming that one now has a new car, determine a replacement policy that minimizes the net costs of owning and operating a car for the next six years.

TABLE 4

Age of Car (Years)	Resale Value ($)	Operating Cost ($)
1	7,000	300 (year 1)
2	6,000	500 (year 2)
3	4,000	800 (year 3)
4	3,000	1,200 (year 4)
5	2,000	1,600 (year 5)
6	1,000	2,200 (year 6)

6 It costs $40 to buy a telephone from the department store. Assume that I can keep a telephone for at most five years and that the estimated maintenance cost each year of operation is as follows: year 1, $20; year 2, $30; year 3, $40; year 4, $60; year 5, $70. I have just purchased a new telephone. Assuming that a telephone has no salvage value, determine how to minimize the total cost of purchasing and operating a telephone for the next six years.

7 At the beginning of year 1, a new machine must be purchased. The cost of maintaining a machine i years old is given in Table 5.

The cost of purchasing a machine at the beginning of each year is given in Table 6.

There is no trade-in value when a machine is replaced. Your goal is to minimize the total cost (purchase plus maintenance) of having a machine for five years. Determine the years in which a new machine should be purchased.

Group B

8[†] A library must build shelving to shelve 200 4-inch high books, 100 8-inch high books, and 80 12-inch high books.

TABLE 5

Age at Beginning of Year	Maintenance Cost for Next Year ($)
0	38,000
1	50,000
2	97,000
3	182,000
4	304,000

[†]Based on Ravindran (1971).

TABLE 6

Year	Purchase Cost ($)
1	170,000
2	190,000
3	210,000
4	250,000
5	300,000

Each book is 0.5 inch thick. The library has several ways to store the books. For example, an 8-inch high shelf may be built to store all books of height less than or equal to 8 inches, and a 12-inch high shelf may be built for the 12-inch books. Alternatively, a 12-inch high shelf might be built to store all books. The library believes it costs $2,300 to build a shelf and that a cost of $5 per square inch is incurred for book storage. (Assume that the area required to store a book is given by height of storage area times book's thickness.)

Formulate and solve a shortest-path problem that could be used to help the library determine how to shelve the books at minimum cost. (*Hint:* Have nodes 0, 4, 8, and 12, with c_{ij} being the total cost of shelving all books of height $> i$ and $\leq j$ on a single shelf.)

9 A company sells seven types of boxes, ranging in volume from 17 to 33 cubic feet. The demand and size of each box is given in Table 7. The variable cost (in dollars) of producing each box is equal to the box's volume. A fixed cost of $1,000 is incurred to produce any of a particular box. If the company desires, demand for a box may be satisfied by a box of larger size. Formulate and solve a shortest-path problem whose solution will minimize the cost of meeting the demand for boxes.

10 Explain how by solving a single transshipment problem you can find the shortest path from node 1 in a network to *each other node* in the network.

TABLE 7

	Box						
	1	2	3	4	5	6	7
Size	33	30	26	24	19	18	17
Demand	400	300	500	700	200	400	200

8.3 Maximum-Flow Problems

Many situations can be modeled by a network in which the arcs may be thought of as having a capacity that limits the quantity of a product that may be shipped through the arc. In these situations, it is often desired to transport the maximum amount of flow from a starting point (called the **source**) to a terminal point (called the **sink**). Such problems are called **maximum-flow problems.** Several specialized algorithms exist to solve maximum-flow problems. In this section, we begin by showing how linear programming can be used to solve a maximum-flow problem. Then we discuss the Ford–Fulkerson (1962) method for solving maximum-flow problems.

LP Solution of Maximum-Flow Problems

EXAMPLE 3 Maximum Flow

Sunco Oil wants to ship the maximum possible amount of oil (per hour) via pipeline from node *so* to node *si* in Figure 6. On its way from node *so* to node *si*, oil must pass through some or all of stations 1, 2, and 3. The various arcs represent pipelines of different diameters. The maximum number of barrels of oil (millions of barrels per hour) that can be pumped through each arc is shown in Table 8. Each number is called an **arc capacity.** Formulate an LP that can be used to determine the maximum number of barrels of oil per hour that can be sent from *so* to *si*.

Solution Node *so* is called the *source* node because oil flows out of it but no oil flows into it. Analogously, node *si* is called the *sink* node because oil flows into it and no oil flows out of it. For reasons that will soon become clear, we have added an artificial arc a_0 from the sink to the source. The flow through a_0 is not actually oil, hence the term **artificial arc.**

To formulate an LP that will yield the maximum flow from node *so* to *si*, we observe that Sunco must determine how much oil (per hour) should be sent through arc (i, j). Thus, we define

$$x_{ij} = \text{millions of barrels of oil per hour that will pass through arc } (i,j) \text{ of pipeline}$$

As an example of a possible flow (termed a *feasible flow*), consider the flow indentified by the numbers in parentheses in Figure 6.

$$x_{so,1} = 2, \quad x_{13} = 0, \quad x_{12} = 2, \quad x_{3,si} = 0, \quad x_{2,si} = 2, \quad x_{si,so} = 2, \quad x_{so,2} = 0$$

FIGURE 6
Network for Sunco Oil

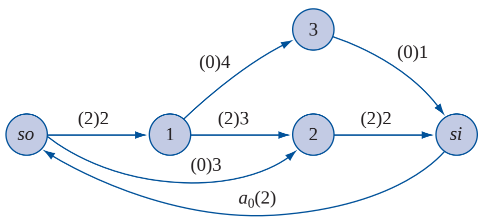


TABLE 8
Arc Capacities for Sunco Oil

Arc	Capacity
(*so*, 1)	2
(*so*, 2)	3
(1, 2)	3
(1, 3)	4
(3, *si*)	1
(2, *si*)	2

For a flow to be feasible, it must have two characteristics:

$$0 \leq \text{flow through each arc} \leq \text{arc capacity} \tag{1}$$

and

$$\text{Flow into node } i = \text{flow out of node } i \tag{2}$$

We assume that no oil gets lost while being pumped through the network, so at each node, a feasible flow must satify (2), the *conservation-of-flow* constraint. The introduction of the artificial arc a_0 allows us to write the conservation-of-flow constraint for the source and sink.

If we let x_0 be the flow through the artificial arc, then conservation of flow implies that x_0 = total amount of oil entering the sink. Thus, Sunco's goal is to maximize x_0 subject to (1) and (2):

$$\begin{array}{llll}
\max z = x_0 & & & \\
\text{s.t.} & & x_{so,1} \leq 2 & \text{(Arc capacity constraints)} \\
& & x_{so,2} \leq 3 & \\
& & x_{12} \leq 3 & \\
& & x_{2,si} \leq 2 & \\
& & x_{13} \leq 4 & \\
& & x_{3,si} \leq 1 & \\
& & x_0 = x_{so,1} + x_{so,2} & \text{(Node } so \text{ flow constraint)} \\
& & x_{so,1} = x_{12} + x_{13} & \text{(Node 1 flow constraint)} \\
& x_{so,2} + x_{12} & = x_{2,si} & \text{(Node 2 flow constraint)} \\
& x_{13} & = x_{3,si} & \text{(Node 3 flow constraint)} \\
& x_{3,si} + x_{2,si} & = x_0 & \text{(Node } si \text{ flow constraint)} \\
& & x_{ij} \geq 0 &
\end{array}$$

One optimal solution to this LP is $z = 3$, $x_{so,1} = 2$, $x_{13} = 1$, $x_{12} = 1$, $x_{so,2} = 1$, $x_{3,si} = 1$, $x_{2,si} = 2$, $x_0 = 3$. Thus, the maximum possible flow of oil from node *so* to *si* is 3 million barrels per hour, with 1 million barrels each sent via the following paths: *so*–1–2–*si*, *so*–1–3–*si*, and *so*–2–*si*.

The linear programming formulation of maximum-flow problems is a special case of the minimum-cost network flow problem (MCNFP) discussed in Section 8.5. A generalization of the transportation simplex (known as the *network simplex*) can be used to solve MCNFPs.

Before discussing the Ford–Fulkerson method for solving maximum-flow problems, we give two examples for situations in which a maximum-flow problem might arise.

EXAMPLE 4 Airline Maximum-Flow

Fly-by-Night Airlines must determine how many connecting flights daily can be arranged between Juneau, Alaska, and Dallas, Texas. Connecting flights must stop in Seattle and then stop in Los Angeles or Denver. Because of limited landing space, Fly-by-Night is limited to making the number of daily flights between pairs of cities shown in Table 9. Set up a maximum-flow problem whose solution will tell the airline how to maximize the number of connecting flights daily from Juneau to Dallas.

TABLE 9
Arc Capacities for Fly-by-Night Airlines

Cities	Maximum Number of Daily Flights
Juneau–Seattle (*J*, *S*)	3
Seattle–L.A. (*S*, *L*)	2
Seattle–Denver (*S*, *De*)	3
L.A.–Dallas (*L*, *D*)	1
Denver–Dallas (*De*, *D*)	2

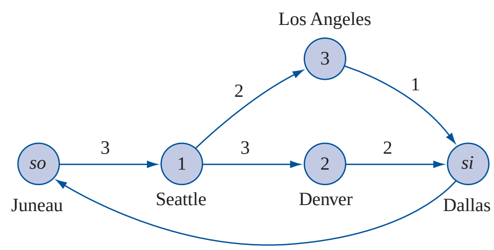


FIGURE 7
Network for Fly-by-Night Airlines

Solution The appropriate network is given in Figure 7. Here the capacity of arc (i, j) is the maximum number of daily flights between city i and city j. The optimal solution to this maximum flow problem is $z = x_0 = 3$, $x_{J,S} = 3$, $x_{S,L} = 1$, $x_{S,De} = 2$, $x_{L,D} = 1$, $x_{De,D} = 2$. Thus, Fly-by-Night can send three flights daily connecting Juneau and Dallas. One flight connects via Juneau–Seattle–L.A.–Dallas, and two flights connect via Juneau–Seattle–Denver–Dallas.

EXAMPLE 5 Matchmaking

Five male and five female entertainers are at a dance. The goal of the matchmaker is to match each woman with a man in a way that maximizes the number of people who are matched with compatible mates. Table 10 describes the compatibility of the entertainers. Draw a network that makes it possible to represent the problem of maximizing the number of compatible pairings as a maximum-flow problem.

Solution Figure 8 is the appropriate network. In Figure 8, there is an arc with capacity 1 joining the source to each man, an arc with capacity 1 joining each pair of compatible mates, and an arc with capacity 1 joining each woman to the sink. The maximum flow in this network is the number of compatible couples that can be created by the matchmaker. For ex-

TABLE 10
Compatibilities for Matching

	Loni Anderson	Meryl Streep	Katharine Hepburn	Linda Evans	Victoria Principal
Kevin Costner	—	C	—	—	—
Burt Reynolds	C	—	—	—	—
Tom Selleck	C	C	—	—	—
Michael Jackson	C	C	—	—	C
Tom Cruise	—	—	C	C	C

Note: C indicates compatibility.

1091.ch08 5/13/03 1:22 PM Page 423

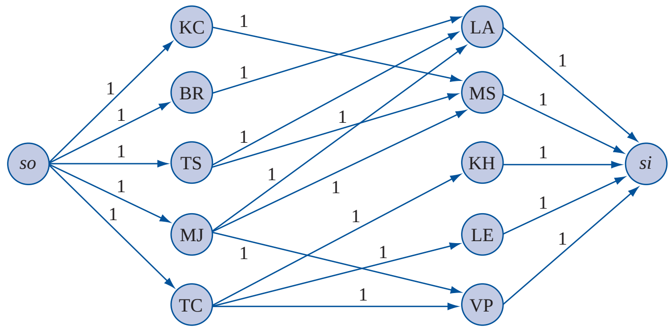


FIGURE 8
Network for Matchmaker

ample, if the matchmaker pairs KC and MS, BR and LA, MJ and VP, and TC and KH, a flow of 4 from source to sink would be obtained. (This turns out to be a maximum flow for the network.)

To see why our network representation correctly models the matchmaker's problem, note that because the arc joining each woman to the sink has a capacity of 1, conservation of flow ensures that each woman will be matched with at most one man. Similarly, because each arc from the source to a man has a capacity of 1, each man can be paired with at most one woman. Because arcs do not exist between noncompatible mates, we can be sure that a flow of *k* units from source to sink represents an assignment of men to women in which *k* compatible couples are created.

Solving Maximum-Flow Problems with LINGO

Maxflow.lng

The maximum flow in a network can be found using LINDO, but LINGO greatly lessens the effort needed to communicate the necessary information to the computer. The following LINGO program (in the file Maxflow.lng) can be used to find the maximum flow from source to sink in Figure 6.

```
MODEL:
  1]SETS:
  2]NODES/1..5/;
  3]ARCS(NODES,NODES)/1,2  1,3  2,3  2,4  3,5  4,5  5,1/
  4]:CAP,FLOW;
  5]ENDSETS
  6]MAX=FLOW (5,1);
  7]@FOR(ARCS(I,J):FLOW(I,J)<CAP(I,J));
  8]@FOR(NODES(I):@SUM(ARCS(J,I):FLOW(J,I))
  9]=@SUM(ARCS(I,J):FLOW(I,J)));
 10]DATA:
 11]CAP=2,3,3,4,2,1,1000;
 12]ENDDATA
END
```

If some nodes are identified by numbers, then LINGO will not allow you to identify other nodes with names involving letters. Thus, we have identified node 1 in line 2 with node *so* in Figure 6 and node 5 in line 2 with node *si*. Also nodes 1, 2, and 3 in Figure 6 correspond to nodes 2, 3, and 4, respectively, in line 2 of our LINGO program. Thus, line 2 defines the nodes of the flow network. In line 3, we define the arcs of the network by listing them (separated by spaces). For example, 1, 2 represents the arc from the source to node 1 in Figure 6 and 5,1 is the artificial arc. In line 4, we indicate that an arc capacity and a flow are associated with each arc. Line 5 ends the definition of the relevant sets.

In line 6, we indicate that our objective is to maximize the flow through the artificial arc (this equals the flow into the sink). Line 7 specifies the arc capacity constraints; for each arc, the flow through the arc cannot exceed the arc's capacity. Lines 8 and 9 create the conservation of flow constraints. For each node I, they ensure that the flow into node I equals the flow out of node I.

Line 10 begins the DATA section. In line 11, we input the arc capacities. Note that we have given the artificial arc a large capacity of 1,000. Line 12 ends the DATA section and the **END** statement ends the program. Typing GO yields the solution, a maximum flow of 3 previously described. The values of the variable FLOW(I,J) give the flow through each arc.

Note that this program can be used to find the maximum flow in any network. Begin by listing the network's nodes in line 2. Then list the network's arcs in line 3. Finally, list the capacity of each arc in the network in line 11, and you are ready to find the maximum flow in the network!

The Ford–Fulkerson Method for Solving Maximum-Flow Problems

We assume that a feasible flow has been found (letting the flow in each arc equal zero gives a feasible flow), and we turn our attention to the following important questions:

Question 1 Given a feasible flow, how can we tell if it is an optimal flow (that is, maximizes x_0)?

Question 2 If a feasible flow is nonoptimal, how can we modify the flow to obtain a new feasible flow that has a larger flow from the source to the sink?

First, we answer question 2. We determine which of the following properties is possessed by each arc in the network:

Property 1 The flow through arc (i, j) is below the capacity of arc (i, j). In this case, the flow through arc (i, j) can be increased. For this reason, we let I represent the set of arcs with this property.

Property 2 The flow in arc (i, j) is positive. In this case, the flow through arc (i, j) can be reduced. For this reason, we let R be the set of arcs with this property.

As an illustration of the definitions of I and R, consider the network in Figure 9. The arcs in this figure may be classified as follows: (*so*, 1) is in I and R; (*so*, 2) is in I; (1, *si*) is in R; (2, *si*) is in I; and (2, 1) is in I.

We can now describe the Ford–Fulkerson labeling procedure used to modify a feasible flow in an effort to increase the flow from the source to the sink.

Step 1 Label the source.

Step 2 Label nodes and arcs (except for arc a_0) according to the following rules: (1) If node x is labeled, then node y is unlabeled and arc (x, y) is a member of I; then label node y and arc (x, y). In this case, arc (x, y) is called a **forward arc.** (2) If node y is unlabeled, node x is labeled and arc (y, x) is a member of R; label node y and arc (y, x). In this case, (y, x) is called a **backward arc.**

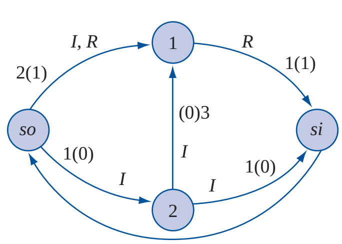


FIGURE 9
Illustration of *I* and *R* arcs

Step 3 Continue this labeling process until the sink has been labeled or until no more vertices can be labeled.

If the labeling process results in the sink being labeled, then there will be a chain of labeled arcs (call it C) leading from the source to the sink. By adjusting the flow of the arcs in C, we can maintain a feasible flow and increase the total flow from source to sink. To see this, observe that C must consist of one of the following:

Case 1 C consists entirely of forward arcs.

Case 2 C contains both forward and backward arcs.[†]

In each case, we can obtain a new feasible flow that has a larger flow from source to sink than the current feasible flow. In Case 1, the chain C consists entirely of forward arcs. For each forward arc in C, let $i(x, y)$ be the amount by which the flow in arc (x, y) can be increased without violating the capacity constraint for arc (x, y) . Let

$$k = \min_{(x,\, y) \in C} i(x, y)$$

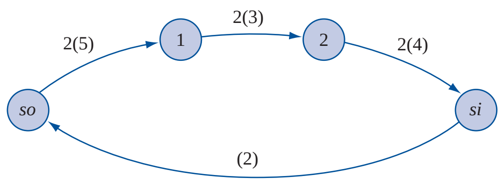
Then $k > 0$. To create a new flow, increase the flow through each arc in C by k units. No capacity constraints are violated, and conservation of flow is still maintained. Thus, the new flow is feasible, and the new feasible flow will transport k more units from source to sink than does the current feasible flow.

We use Figure 10 to illustrate Case 1. Currently, 2 units are being transported from source to sink. The labeling procedure results in the sink being labeled by the chain $C = (so, 1) - (1, 2) - (2, si)$. Each arc is in I, and $i(so, 1) = 5 - 2 = 3$; $i(1, 2) = 3 - 2 = 1$; and $i(2, si) = 4 - 2 = 2$. Hence, $k = \min(3, 1, 2) = 1$. Thus, an improved feasible flow can be obtained by increasing the flow on each arc in C by 1 unit. The resulting flow transports 3 units from source to sink (see Figure 11).

In Case 2, the chain C leading from the source to the sink contains both backward and forward arcs. For each backward arc in C, let $r(x, y)$ be the amount by which the flow through arc (x, y) can be reduced. Also define

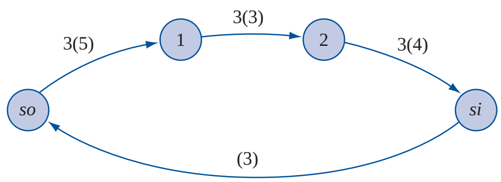
$$k_1 = \min_{x,\, y \in C \cap R} r(x, y) \quad \text{and} \quad k_2 = \min_{x,\, y \in C \cap I} i(x, y)$$

FIGURE 10
Illustration of Case 1 of Labeling Method



Flow from source to sink = 2
Chain is (*so*, 1) – (1, 2) – (2, *si*)

FIGURE 11
Improved Flow from Source to Sink: Case 1



Flow from source to sink = 3

[†]Because we exclude arc a_0 from the labeling procedure, no chain made entirely of backward arcs can lead from source to sink.

Of course, both k_1 and k_2 and min (k_1,k_2) are > 0. To increase the flow from source to sink (while maintaining a feasible flow), decrease the flow in all of C's backward arcs by min (k_1, k_2) and increase the flow in all of C's forward arcs by $\min(k_1, k_2)$. This will maintain conservation of flow and ensure that no arc capacity constraints are violated. Because the last arc in C is a forward arc leading into the sink, we have found a new feasible flow and have increased the total flow into the sink by $\min(k_1, k_2)$. We now adjust the flow in the arc a_0 to maintain conservation of flow. To illustrate Case 2, suppose we have found the feasible flow in Figure 12. For this flow, $(so, 1) \in R$; $(so, 2) \in I$; $(1, 3) \in I$; $(1, 2) \in I$ and R; $(2, si) \in R$; and $(3, si) \in I$.

We begin by labeling arc $(so, 2)$ and node 2 (thus $(so, 2)$ is a forward arc). Then we label arc (1, 2) and node 1. Arc (1, 2) is a backward arc, because node 1 was unlabeled before we labeled arc (1, 2), and arc (1, 2) is in R. Nodes so, 1, and 2 are labeled, so we can label arc (1, 3) and node 3. [Arc (1, 3) is a forward arc, because node 3 has not yet been labeled.] Finally we label arc $(3, si)$ and node si. Arc $(3, si)$ is a forward arc, because node si has not yet been labeled. We have now labeled the sink via the chain $C = (so, 2) - (1, 2) - (1, 3) - (3, si)$. With the exception of arc (1, 2), all arcs in the chain are forward arcs. Because $i(so, 2) = 3$; $i(1, 3) = 4$; $i(3, si) = 1$; and $r(1, 2) = 2$, we have

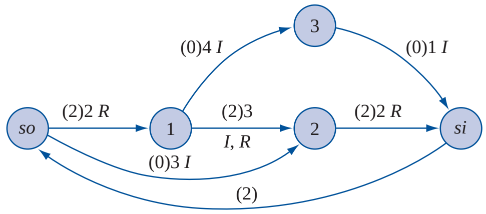
$$\min_{(x,\, y)\in C\cap R} r(x, y) = 2 \qquad \text{and} \qquad \min_{(x,\, y)\in C\cap I} i(x, y) = 1$$

Thus, we can increase the flow on all forward arcs in C by 1 and decrease the flow in all backward arcs by 1. The new result, pictured in Figure 13, has increased the flow from source to sink by 1 unit (from 2 to 3). We accomplish this by diverting 1 unit that was transported through the arc (1, 2) to the path 1–3–si. This enabled us to transport an extra unit from source to sink via the path so–2–si. Observe that the concept of a backward arc was needed to find this improved flow.

If the sink cannot be labeled, then the current flow is optimal. The proof of this fact relies on the concept of a *cut* for a network.

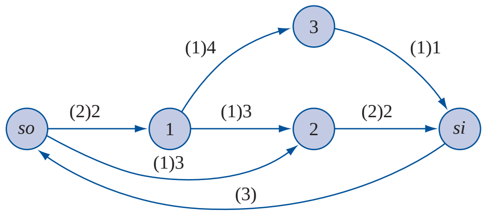
DEFINITION ■ Choose any set of nodes V' that contains the sink but does not contain the source. Then the set of arcs (i, j) with i not in V' and j a member of V' is a **cut** for the network. ■

FIGURE 12
Illustration of Case 2 of Labeling Method



Flow from source to sink = 2
Chain is $(so, 2) - (1, 2) - (1, 3) - (3, si)$

FIGURE 13
Improved Flow from Source to Sink: Case 2



Flow from source to sink = 3

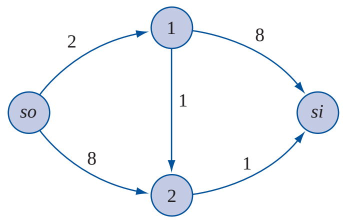


FIGURE 14
Example of a Cut

$V' = [1, si]$ yields cut $[(so, 1), (2, si)]$

DEFINITION ■ The **capacity** of a cut is the sum of the capacities of the arcs in the cut. ■

In short, a cut is a set of arcs whose removal from the network makes it impossible to travel from the source to the sink. A network may have many cuts. For example, in the network in Figure 14, $V' = \{1, si\}$ yields the cut containing the arcs $(so, 1)$ and $(2, si)$, which has capacity $2 + 1 = 3$. The set $V' = \{1, 2, si\}$ yields the cut containing the arcs $(so, 1)$ and $(so, 2)$, which has capacity $2 + 8 = 10$.

Lemma 1 and Lemma 2 indicate the connection between cuts and maximum flows.

LEMMA 1

The flow from source to sink for any feasible flow is less than or equal to the capacity of *any* cut.

Proof Consider an arbitrary cut specified by a set of nodes V' that contains the sink but does not contain the source. Let V be all other nodes in the network. Also let x_{ij} be the flow in arc (i, j) for any feasible flow and f be the flow from source to sink for this feasible flow. Summing the flow balance equations (flow out of node i − flow into node $i = 0$) over all nodes i in V, we find that the terms involving arcs (i, j) having i and j both members of V will cancel, and we obtain

$$\sum_{\substack{i \in V; \\ j \in V'}} x_{ij} - \sum_{\substack{i \in V'; \\ j \in V}} x_{ij} = f \tag{3}$$

Now the first sum in (3) equals the capacity of the cut. Each x_{ij} is nonnegative, so we see that $f \leq$ capacity of the cut, which is the desired result.

Lemma 1 is analogous to the weak duality result discussed in Chapter 6. From Lemma 1, we see that the capacity of any cut is an upper bound for the maximum flow from source to sink. Thus, if we can find a feasible flow and a cut for which the flow from source to sink equals the capacity of the cut, then we have found the maximum flow from source to sink.

Suppose that we find a feasible flow and cannot label the sink. Let CUT be the cut corresponding to the set of unlabeled nodes.

LEMMA 2

If the sink cannot be labeled, then

$$\text{Capacity of CUT} = \text{current flow from source to sink}$$

Proof Let V' be the set of unlabeled nodes and V be the set of labeled nodes. Consider an arc (i, j) such that i is in V and j is in V'. Then we know that x_{ij} = capacity of arc (i, j) must hold; otherwise, we could label node j (via a forward arc) and node j would not be in V'. Now consider an arc (i, j) such that i is in V' and j is in V. Then $x_{ij} = 0$ must hold; otherwise, we could label node i (via a backward arc) and node i would not be in V'. Now (3) shows that the current flow must satisfy

$$\text{Capacity of CUT} = \text{current flow from source to sink}$$

which is the desired result.

From the remarks following Lemma 1, when the sink cannot be labeled, the maximum flow from source to sink has been obtained.

Summary and Illustration of the Ford–Fulkerson Method

Step 1 Find a feasible flow (setting each arc's flow to zero will do).

Step 2 Using the labeling procedure, try to label the sink. If the sink cannot be labeled, then the current feasible flow is a maximum flow; if the sink is labeled, then go on to step 3.

Step 3 Using the method previously described, adjust the feasible flow and increase the flow from the source to the sink. Return to step 2.

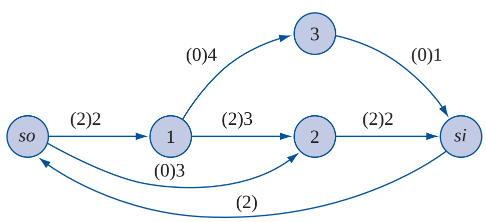
To illustrate the Ford–Fulkerson method, we find the maximum flow from source to sink for Sunco Oil, Example 3 (see Figure 6). We begin by letting the flow in each arc equal zero. We then try to label the sink—label the source, and then arc (*so*, 1) and node 1; then label arc (1, 2) and node 2; finally, label arc (2, *si*) and node *si*. Thus, $C =$ (so, 1)–(1, 2)–(2, *si*). Each arc in C is a forward arc, so we can increase the flow through each arc in C by min $(2, 3, 2) = 2$ units. The resulting flow is pictured in Figure 15.

As we saw previously (Figure 12), we can label the sink by using the chain $C =$ (*so*, 2)–(1, 2)–(1, 3)–(3, *si*). We can increase the flow through the forward arcs (*so*, 2), (1, 3), and (3, *si*) by 1 unit and decrease the flow through the backward arc (1, 2) by 1 unit. The resulting flow is pictured in Figure 16. It is now impossible to label the sink. Any attempt to label the sink must begin by labeling arc (*so*, 2) and node 2; then we could label arc (1, 2) and arc (1, 3). But there is no way to label the sink.

We can verify that the current flow is maximal by finding the capacity of the cut corresponding to the set of unlabeled vertices (in this case, *si*). The cut corresponding to *si* is the set of arcs (2, *si*) and (3, *si*), with capacity $2 + 1 = 3$. Thus, Lemma 1 implies that any feasible flow can transport at most 3 units from source to sink. Our current flow transports 3 units from source to sink, so it must be an optimal flow.

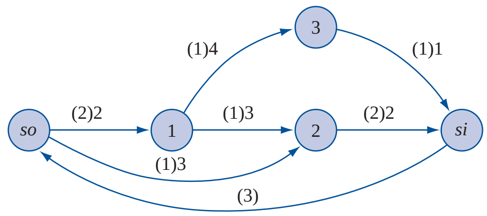
Another example of the Ford–Fulkerson method is given in Figure 17. Note that without the concept of a backward arc, we could not have obtained the maximum flow of 7

FIGURE 15
Network for Sunco Oil (Increased Flow)



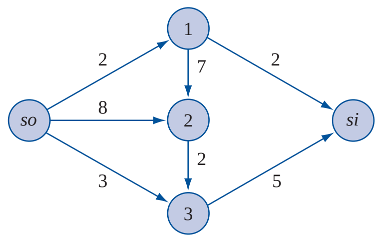
Flow from source to sink = 2
Label sink by (*so*, 2) – (1, 2) – (1, 3) – (3, *si*)

FIGURE 16
Network for Sunco Oil (Optimal Flow)

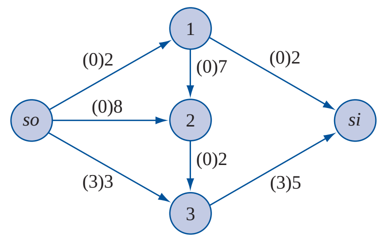


Flow from source to sink = 3
Since sink cannot be labeled, this is an optimal flow

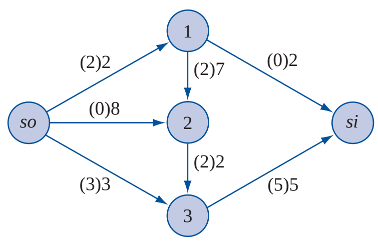
FIGURE 17
Example of Ford–Fulkerson Method



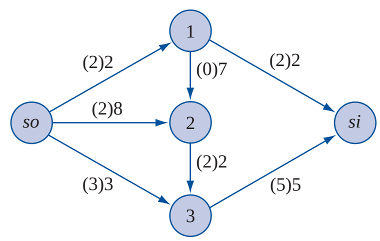
a Original network



b Label sink by *so* – 3 – *si* (adds 3 units of flow using only forward arcs)



c Label sink by *so* – 1 – 2 – 3 – *si* (adds 2 units of flow using only forward arcs)



d Label sink by *so* – 2 – 1 – *si* (adds 2 units of flow using backward arc (1, 2); maximum flow of 7 has been obtained

units from source to sink. The minimum cut (with capacity 7, of course) corresponds to nodes 1, 3, and *si* and consists of arcs (*so*, 1), (so, 3) and (2, 3).

PROBLEMS

Group A

1–3 Figures 18–20 show the networks for Problems 1–3. Find the maximum flow from source to sink in each network. Find a cut in the network whose capacity equals the maximum flow in the network. Also, set up an LP that could be used to determine the maximum flow in the network.

FIGURE 18
Network for Problem 1

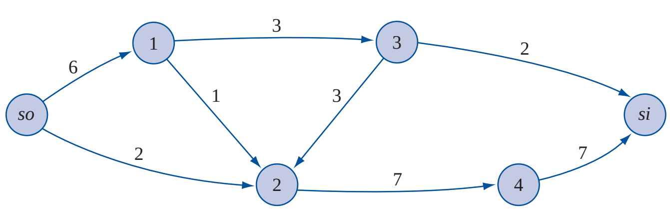


FIGURE 19
Network for Problem 2

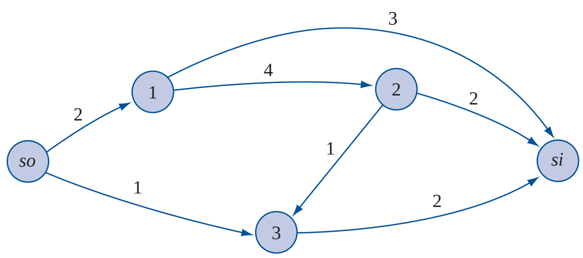


FIGURE 20
Nework for Problem 3

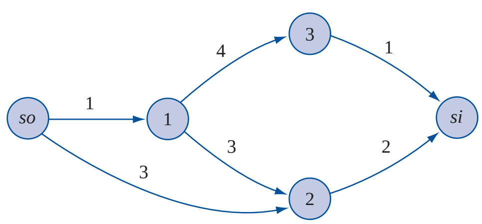


FIGURE 21
Network for Problem 4

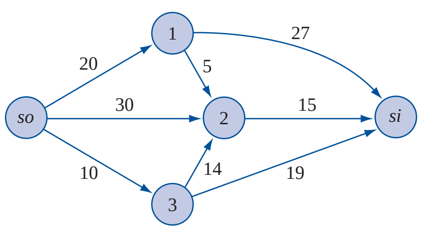
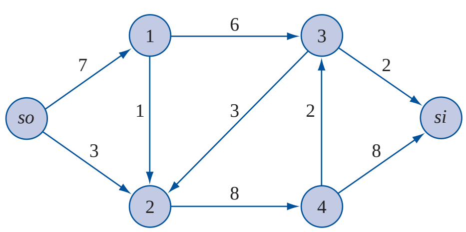


FIGURE 22
Network for Problem 5



4–5 For the networks in Figures 21 and 22, find the maximum flow from source to sink. Also find a cut whose capacity equals the maximum flow in the network.

6 Seven types of packages are to be delivered by five trucks. There are three packages of each type, and the capacities of the five trucks are 6, 4, 5, 4, and 3 packages, respectively. Set up a maximum-flow problem that can be used to determine whether the packages can be loaded so that no truck carries two packages of the same type.

7 Four workers are available to perform jobs 1–4. Unfortunately, three workers can do only certain jobs: worker 1, only job 1; worker 2, only jobs 1 and 2; worker 3, only job 2; worker 4, any job. Draw the network for the maximum-flow problem that can be used to determine whether all jobs can be assigned to a suitable worker.

8 The Hatfields, Montagues, McCoys, and Capulets are going on their annual family picnic. Four cars are available to transport the families to the picnic. The cars can carry the following number of people: car 1, four; car 2, three; car 3, three; and car 4, four. There are four people in each family, and no car can carry more than two people from any one family. Formulate the problem of transporting the maximum possible number of people to the picnic as a maximum-flow problem.

9–10 For the networks in Figures 23 and 24, find the maximum flow from source to sink. Also find a cut whose capacity equals the maximum flow in the network.

Group B

11 Suppose a network contains a finite number of arcs and the capacity of each arc is an integer. Explain why the Ford–Fulkerson method will find the maximum flow in the finite number of steps. Also show that the maximum flow from source to sink will be an integer.

12 Consider a network flow problem with several sources and several sinks in which the goal is to maximize the total flow into the sinks. Show how such a problem can be converted into a maximum-flow problem having only a single source and a single sink.

FIGURE 23

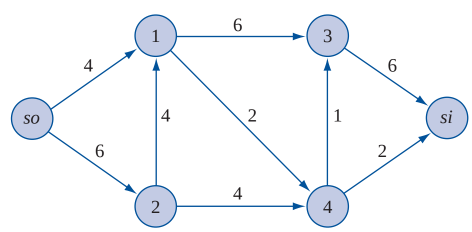
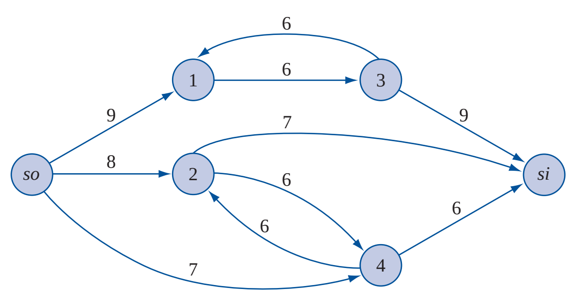


FIGURE 24



1091.ch08 5/13/03 1:23 PM Page 431

13 Suppose the total flow into a node of a network is restricted to 10 units or less. How can we represent this restriction via an arc capacity constraint? (This still allows us to use the Ford–Fulkerson method to find the maximum flow.)

14 Suppose as many as 300 cars per hour can travel between any two of the cities 1, 2, 3, and 4. Set up a maximum-flow problem that can be used to determine how many cars can be sent in the next two hours from city 1 to city 4. (*Hint:* Have portions of the network represent $t = 0$, $t = 1$, and $t = 2$.)

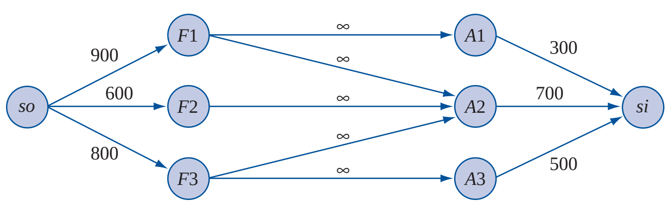
15 Fly-by-Night Airlines is considering flying three flights. The revenue from each flight and the airports used by each flight are shown in Table 11. When Fly-by-Night uses an airport, the company must pay the following landing fees (independent of the number of flights using the airport): airport 1, \$300; airport 2, \$700; airport 3, \$500. Thus, if flights 1 and 3 are flown, a profit of $900 + 800 - 300 - 700 - 500 =$ \$200 will be earned. Show that for the network in Figure 25 (maximum profit) = (total revenue from all flights) − (capacity of minimal cut). Explain how this result can be used to help Fly-by-Night maximize profit (even if it has hundreds of possible flights). (*Hint:* Consider any set of flights F (say, flights 1 and 3). Consider the cut corresponding to the sink, the nodes associated with the flights not in F, and the nodes associated with the airports not used by F. Show that (capacity of this cut) = (revenue from flights not in F) + (costs associated with airports used by F).)

TABLE 11

Flight	Revenue (\$)	Airport Used
1	900	1 and 2
2	600	2
3	800	2 and 3

16 During the next four months, a construction firm must complete three projects. Project 1 must be completed within three months and requires 8 months of labor. Project 2 must be completed within four months and requires 10 months of labor. Project 3 must be completed at the end of two months and requires 12 months of labor. Each month, 8 workers are available. During a given month, no more than 6 workers can work on a single job. Formulate a maximum-flow problem that could be used to determine whether all three projects can be completed on time. (*Hint:* If the maximum flow in the network is 30, then all projects can be completed on time.)

FIGURE 25
Network for Problem 15



8.4 CPM and PERT

Network models can be used as an aid in scheduling large complex projects that consist of many activities. If the duration of each activity is known with certainty, then the **critical path method (CPM)** can be used to determine the length of time required to complete a project. CPM also can be used to determine how long each activity in the project can be delayed without delaying the completion of the project. CPM was developed in the late 1950s by researchers at DuPont and Sperry Rand.

If the duration of the activities is not known with certainty, the Program Evaluation and Review Technique (PERT) can be used to estimate the probability that the project will be completed by a given deadline. PERT was developed in the late 1950s by consultants working on the development of the Polaris missile. CPM and PERT were given a major share of the credit for the fact that the Polaris missile was operational two years ahead of schedule.

CPM and PERT have been successfully used in many applications, including:

1 Scheduling construction projects such as office buildings, highways, and swimming pools

2 Scheduling the movement of a 400-bed hospital from Portland, Oregon, to a suburban location

3 Developing a countdown and "hold" procedure for the launching of space flights

4 Installing a new computer system

5 Designing and marketing a new product

6 Completing a corporate merger

7 Building a ship

To apply CPM and PERT, we need a list of the activities that make up the project. The project is considered to be completed when all the activities have been completed. For each activity, there is a set of activities (called the **predecessors** of the activity) that must be completed before the activity begins. A project network is used to represent the precedence relationships between activities. In our discussion, activities will be represented by directed arcs, and nodes will be used to represent the completion of a set of activities. (For this reason, we often refer to the nodes in our project network as **events.**) This type of project network is called an **AOA (activity on arc)** network.†

To understand how an AOA network represents precedence relationships, suppose that activity A is a predecessor of activity B. Each node in an AOA network represents the completion of one or more activities. Thus, node 2 in Figure 26 represents the completion of activity A and the beginning of activity B. Suppose activities A and B must be completed before activity C can begin. In Figure 27, node 3 represents the event that activities A and B are completed. Figure 28 shows activity A as a predecessor of both activities B and C.

Given a list of activities and predecessors, an AOA representation of a project (called a **project network** or **project diagram**) can be constructed by using the following rules:

1 Node 1 represents the start of the project. An arc should lead from node 1 to represent each activity that has no predecessors.

FIGURE 26
Activity A Must Be Completed Before Activity B Can Begin

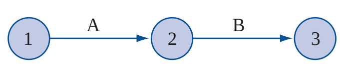


FIGURE 27
Activities A and B Must Be Completed Before Activity C Can Begin

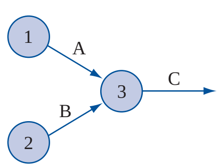
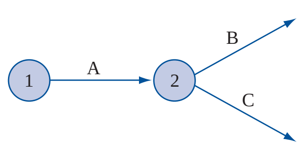


FIGURE 28
Activity A Must Be Completed Before Activities B and C Can Begin



†In an AON (activity on node) project network, the nodes of the network are used to represent activities. See Wiest and Levy (1977) for details.

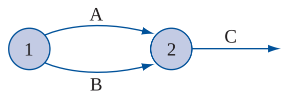


FIGURE 29
Violation of Rule 5

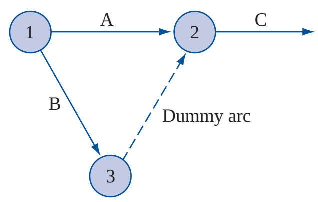


FIGURE 30
Use of Dummy Activity

2 A node (called the **finish node**) representing the completion of the project should be included in the network.

3 Number the nodes in the network so that the node representing the completion of an activity always has a larger number than the node representing the beginning of an activity (there may be more than one numbering scheme that satisfies rule 3).

4 An activity should not be represented by more than one arc in the network.

5 Two nodes can be connected by at most one arc.

To avoid violating rules 4 and 5, it is sometimes necessary to utilize a **dummy activity** that takes zero time. For example, suppose activities A and B are both predecessors of activity C and can begin at the same time. In the absence of rule 5, we could represent this by Figure 29. However, because nodes 1 and 2 are connected by more than one arc, Figure 29 violates rule 5. By using a dummy activity (indicated by a dotted arc), as in Figure 30, we may represent the fact that A and B are both predecessors of C. Figure 30 ensures that activity C cannot begin until both A and B are completed, but it does not violate rule 5. Problem 10 at the end of this section illustrates how dummy activities may be needed to avoid violating rule 4.

Example 6 illustrates a project network.

EXAMPLE 6 Drawing a Project Network

Widgetco is about to introduce a new product (product 3). One unit of product 3 is produced by assembling 1 unit of product 1 and 1 unit of product 2. Before production begins on either product 1 or 2, raw materials must be purchased and workers must be trained. Before products 1 and 2 can be assembled into product 3, the finished product 2 must be inspected. A list of activities and their predecessors and of the duration of each activity is given in Table 12. Draw a project diagram for this project.

TABLE 12
Duration of Activities and Predecessor Relationships for Widgetco

Activity	Predecessors	Duration (Days)
A = train workers	—	6
B = purchase raw materials	—	9
C = produce product 1	A, B	8
D = produce product 2	A, B	7
E = test product 2	D	10
F = assemble products 1 and 2	C, E	12

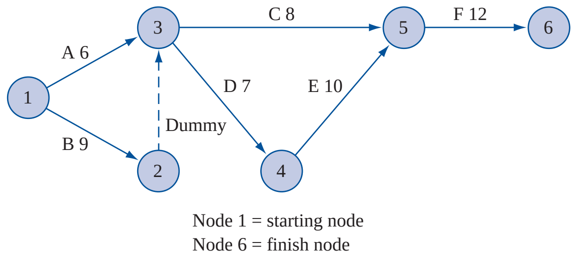


FIGURE 31
Project Diagram for Widgetco

Solution Observe that although we list only C and E as predecessors of F, it is actually true that activities A, B, and D must also be completed before F begins. C cannot begin until A and B are completed, and E cannot begin until D is completed, however, so it is redundant to state that A, B, and D are predecessors of F. Thus, in drawing the project network, we need only be concerned with the immediate predecessors of each activity.

The AOA network for this project is given in Figure 31 (the number above each arc represents activity duration in days). Node 1 is the beginning of the project, and node 6 is the finish node representing completion of the project. The dummy arc (2, 3) is needed to ensure that rule 5 is not violated.

The two key building blocks in CPM are the concepts of early event time (ET) and late event time (LT) for an event.

DEFINITION ■ The **early event time** for node i, represented by $ET(i)$, is the earliest time at which the event corresponding to node i can occur. ■

The **late event time** for node i, represented by $LT(i)$, is the latest time at which the event corresponding to node i can occur without delaying the completion of the project. ■

Computation of Early Event Time

To find the early event time for each node in the project network, we begin by noting that because node 1 represents the start of the project, $ET(1) = 0$. We then compute $ET(2)$, $ET(3)$, and so on, stopping when $ET(\text{finish node})$ has been calculated. To illustrate how $ET(i)$ is calculated, suppose that for the segment of a project network in Figure 32, we have already determined that $ET(3) = 6$, $ET(4) = 8$, and $ET(5) = 10$. To determine $ET(6)$, observe that the earliest time that node 6 can occur is when the activities corresponding to arc (3, 6), (4, 6), and (5, 6) have *all* been completed.

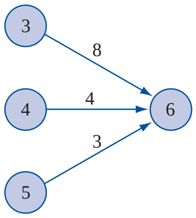


FIGURE 32
Determination of $ET(6)$

$$ET(6) = \max \begin{cases} ET(3) + 8 = 14 \\ ET(4) + 4 = 12 \\ ET(5) + 3 = 13 \end{cases}$$

Thus, the earliest time that node 6 can occur is 14, and $ET(6) = 14$.

From this example, it is clear that computation of $ET(i)$ requires (for $j < i$) knowledge of one or more of the $ET(j)$'s. This explains why we begin by computing the predecessor ETs. In general, if $ET(1), ET(2), \ldots, ET(i-1)$ have been determined, then we compute $ET(i)$ as follows:

Step 1 Find each prior event to node i that is connected by an arc to node i. These events are the **immediate predecessors** of node i.

Step 2 To the ET for each immediate predecessor of the node i add the duration of the activity connecting the immediate predecessor to node i.

Step 3 $ET(i)$ equals the maximum of the sums computed in step 2.

We now compute the $ET(i)$'s for Example 6. We begin by observing that $ET(1) = 0$. Node 1 is the only immediate predecessor of node 2, so $ET(2) = ET(1) + 9 = 9$. The immediate predecessors of node 3 are nodes 1 and 2. Thus,

$$ET(3) = \max \begin{cases} ET(1) + 6 = 6 \\ ET(2) + 0 = 9 \end{cases} = 9$$

Node 4's only immediate predecessor is node 3. Thus, $ET(4) = ET(3) + 7 = 16$. Node 5's immediate predecessors are nodes 3 and 4. Thus,

$$ET(5) = \max \begin{cases} ET(3) + 8 = 17 \\ ET(4) + 10 = 26 \end{cases} = 26$$

Finally, node 5 is the only immediate predecessor of node 6. Thus, $ET(6) = ET(5) + 12 = 38$. Because node 6 represents the completion of the project, we see that the earliest time that product 3 can be assembled is 38 days from now.

It can be shown that $ET(i)$ is the length of the longest path in the project network from node 1 to node i.

Computation of Late Event Time

To compute the $LT(i)$'s, we begin with the finish node and work backward (in descending numerical order) until we determine $LT(1)$. The project in Example 6 can be completed in 38 days, so we know that $LT(6) = 38$. To illustrate how $LT(i)$ is computed for nodes other than the finish node, suppose we are working with a network (Figure 33) for which we have already determined that $LT(5) = 24$, $LT(6) = 26$, and $LT(7) = 28$. In this situation, how can we compute $LT(4)$? If the event corresponding to node 4 occurs after $LT(5) - 3$, node 5 will occur after $LT(5)$, and the completion of the project will be delayed.

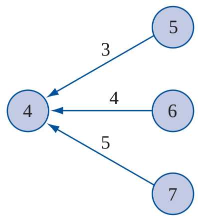


FIGURE 33
Computation of $LT(4)$

Similarly, if node 4 occurs after $LT(6) - 4$ or if node 4 occurs after $LT(7) - 5$, the completion of the project will be delayed. Thus,

$$LT(4) = \min \begin{cases} LT(5) - 3 = 21 \\ LT(6) - 4 = 22 = 21 \\ LT(7) - 5 = 23 \end{cases}$$

In general, if $LT(j)$ is known for $j > i$, we can find $LT(i)$ as follows:

Step 1 Find each node that occurs after node i and is connected to node i by an arc. These events are the **immediate successors** of node i.

Step 2 From the LT for each immediate successor to node i, subtract the duration of the activity joining the successor the node i.

Step 3 $LT(i)$ is the smallest of the differences determined in step 2.

We now compute the $LT(i)$'s for Example 6. Recall that $LT(6) = 38$. Because node 6 is the only immediate successor of node 5, $LT(5) = LT(6) - 12 = 26$. Node 4's only immediate successor is node 5. Thus, $LT(4) = LT(5) - 10 = 16$. Nodes 4 and 5 are immediate successors of node 3. Thus,

$$LT(3) = \min \begin{cases} LT(4) - 7 = 9 \\ LT(5) - 8 = 18 \end{cases}$$

Node 3 is the only immediate successor of node 2. Thus, $LT(2) = LT(3) - 0 = 9$. Finally, node 1 has nodes 2 and 3 as immediate successors. Thus,

$$LT(1) = \min \begin{cases} LT(3) - 6 = 3 \\ LT(2) - 9 = 0 \end{cases}$$

Table 13 summarizes our computations for Example 6. If $LT(i) = ET(i)$, any delay in the occurrence of node i will delay the completion of the project. For example, because $LT(4) = ET(4)$, any delay in the occurrence of node 4 will delay the completion of the project.

Total Float

Before the project is begun, the duration of an activity is unknown, and the duration of each activity used to construct the project network is just an estimate of the activity's actual completion time. The concept of total float of an activity can be used as a measure of how important it is to keep each activity's duration from greatly exceeding our estimate of its completion time.

TABLE 13
ET and *LT* for Widgetco

Node	*ET(i)*	*LT(i)*
1	0	0
2	9	9
3	9	9
4	16	16
5	26	26
6	38	38

DEFINITION ■ For an arbitrary arc representing activity (i, j), the **total float,** represented by $TF(i, j)$, of the activity represented by (i, j) is the amount by which the starting time of activity (i, j) could be delayed beyond its earliest possible starting time without delaying the completion of the project (assuming no other activities are delayed). ■

Equivalently, the total float of an activity is the amount by which the duration of the activity can be increased without delaying the completion of the project.

If we define t_{ij} to be the duration of activity (i, j), then $TF(i, j)$ can easily be expressed in terms of $LT(j)$ and $ET(i)$. Activity (i, j) begins at node i. If the occurrence of node i, or the duration of activity (i, j), is delayed by k time units, then activity (i, j) will be completed at time $ET(i) + k + t_{ij}$. Thus, the completion of the project will not be delayed if

$$ET(i) + k + t_{ij} \leq LT(j) \qquad \text{or} \qquad k \leq LT(j) - ET(i) - t_{ij}$$

Therefore,

$$TF(i, j) = LT(j) - ET(i) - t_{ij}$$

For Example 6, the $TF(i, j)$ are as follows:

Activity B: $TF(1, 2) = LT(2) - ET(1) - 9 = 0$
Activity A: $TF(1, 3) = LT(3) - ET(1) - 6 = 3$
Activity D: $TF(3, 4) = LT(4) - ET(3) - 7 = 0$
Activity C: $TF(3, 5) = LT(5) - ET(3) - 8 = 9$
Activity E: $TF(4, 5) = LT(5) - ET(4) - 10 = 0$
Activity F: $TF(5, 6) = LT(6) - ET(5) - 12 = 0$
Dummy activity: $TF(2, 3) = LT(3) - ET(2) - 0 = 0$

Finding a Critical Path

If an activity has a total float of zero, then any delay in the start of the activity (or the duration of the activity) will delay the completion of the project. In fact, increasing the duration of an activity by Δ days will increase the length of the project by Δ days. Such an activity is critical to the completion of the project on time.

DEFINITION ■ Any activity with a total float of zero is a **critical activity.** ■

A path from node 1 to the finish node that consists entirely of critical activities is called a **critical path.** ■

In Figure 31, activities B, D, E, F, and the dummy activity are critical activities and the path 1–2–3–4–5–6 is the critical path (it is possible for a network to have more than one critical path). A critical path in any project network is the longest path from the start node to the finish node (see Problem 2 in Section 8.5).

Any delay in the duration of a critical activity will delay the completion of the project, so it is advisable to monitor closely the completion of critical activities.

Free Float

As we have seen, the total float of an activity can be used as a measure of the flexibility in the duration of an activity. For example, activity A can take up to 3 days longer than its scheduled duration of 6 days without delaying the completion of the project. Another measure of the flexibility available in the duration of an activity is free float.

DEFINITION ■ The **free float** of the activity corresponding to arc (i, j), denoted by $FF(i, j)$, is the amount by which the starting time of the activity corresponding to arc (i, j) (or the duration of the activity) can be delayed without delaying the start of any later activity beyond its earliest possible starting time. ■

Suppose the occurrence of node i, or the duration of activity (i, j), is delayed by k units. Then the earliest that node j can occur is $ET(i) + t_{ij} + k$. Thus, if $ET(i) + t_{ij} + k \leq ET(j)$, or $k \leq ET(j) - ET(i) - t_{ij}$, then node j will not be delayed. If node j is not delayed, then no other activities will be delayed beyond their earliest possible starting times. Therefore,

$$FF(i, j) = ET(j) - ET(i) - t_{ij}$$

For Example 6, the $FF(i, j)$ are as follows:

Activity B: $FF(1, 2) = 9 - 0 - 9 = 0$
Activity A: $FF(1, 3) = 9 - 0 - 6 = 3$
Activity D: $FF(3, 4) = 16 - 9 - 7 = 0$
Activity C: $FF(3, 5) = 26 - 9 - 8 = 9$
Activity E: $FF(4, 5) = 26 - 16 - 10 = 0$
Activity F: $FF(5, 6) = 38 - 26 - 12 = 0$

For example, because the free float for activity C is 9 days, a delay in the start of activity C (or in the occurrence of node 3) or a delay in the duration of activity C of more than 9 days will delay the start of some later activity (in this case, activity F).

Using Linear Programming to Find a Critical Path

Although the previously described method for finding a critical path in a project network is easily programmed on a computer, linear programming can also be used to determine the length of the critical path. Define

$$x_j = \text{the time that the event corresponding to node } j \text{ occurs}$$

For each activity (i, j), we know that before node j occurs, node i must occur and activity (i, j) must be completed. This implies that for each arc (i, j) in the project network, $x_j \geq x_i + t_{ij}$. Let F be the node that represents completion of the project. Our goal is to minimize the time required to complete the project, so we use an objective function of $z = x_F - x_1$.

To illustrate how linear programming can be used to find the length of the critical path, we apply the preceding approach to Example 6. The appropriate LP is

$$\begin{array}{lll}
\min z = x_6 - x_1 & & \\
\text{s.t.} \quad x_3 \geq x_1 + 6 & & \text{(Arc (1, 3) constraint)} \\
\quad\quad x_2 \geq x_1 + 9 & & \text{(Arc (1, 2) constraint)} \\
\quad\quad x_5 \geq x_3 + 8 & & \text{(Arc (3, 5) constraint)} \\
\quad\quad x_4 \geq x_3 + 7 & & \text{(Arc (3, 4) constraint)}
\end{array}$$

$$x_5 \geq x_4 + 10 \qquad \text{(Arc (4, 5) constraint)}$$
$$x_6 \geq x_5 + 12 \qquad \text{(Arc (5, 6) constraint)}$$
$$x_3 \geq x_2 \qquad \text{(Arc (2, 3) constraint)}$$
$$\text{All variables urs}$$

An optimal solution to this LP is $z = 38$, $x_1 = 0$, $x_2 = 9$, $x_3 = 9$, $x_4 = 16$, $x_5 = 26$, and $x_6 = 38$. This indicates that the project can be completed in 38 days.

This LP has many alternative optimal solutions. In general, the value of x_i in any optimal solution may assume any value between $ET(i)$ and $LT(i)$. All optimal solutions to this LP, however, will indicate that the length of any critical path is 38 days.

A critical path for this project network consists of a path from the start of the project to the finish in which each arc in the path corresponds to a constraint having a dual price of -1. From the LINDO output in Figure 34, we find, as before, that 1–2–3–4–5–6 is a critical path. For each constraint with a dual price of -1, increasing the duration of the activity corresponding to that constraint by Δ days will increase the duration of the project by Δ days. For example, an increase of Δ days in the duration of activity B will increase the duration of the project by Δ days. This assumes that the current basis remains optimal.

Crashing the Project

In many situations, the project manager must complete the project in a time that is less than the length of the critical path. For instance, suppose Widgetco believes that to have any chance of being a success, product 3 must be available for sale before the competitor's product hits the market. Widgetco knows that the competitor's product is scheduled to hit the market 26 days from now, so Widgetco must introduce product 3 within 25 days. Because the critical path in Example 6 has a length of 38 days, Widgetco will have to expend additional resources to meet the 25-day project deadline. In such a situation, linear programming can often be used to determine the allocation of resources that minimizes the cost of meeting the project deadline.

Suppose that by allocating additional resources to an activity, Widgetco can reduce the duration of any activity by as many as 5 days. The cost per day of reducing the duration of an activity is shown in Table 14. To find the minimum cost of completing the project by the 25-day deadline, define variables A, B, C, D, E, and F as follows:

$$A = \text{number of days by which duration of activity } A \text{ is reduced}$$
$$\vdots \qquad\qquad \vdots$$
$$F = \text{number of days by which duration of activity } F \text{ is reduced}$$
$$x_j = \text{time that the event corresponding to node } j \text{ occurs}$$

Then Widgetco should solve the following LP:

$$\min z = 10A + 20B + 3C + 30D + 40E + 50F$$
$$\text{s.t.} \quad A \leq 5$$
$$B \leq 5$$
$$C \leq 5$$
$$D \leq 5$$
$$E \leq 5$$
$$F \leq 5$$

```
MIN      X6 - X1
 SUBJECT TO
        2) - X1 + X3 >=  6
        3) - X1 + X2 >=  9
        4) - X3 + X5 >=  8
        5) - X3 + X4 >=  7
        6)   X5 - X4 >=  10
        7)   X6 - X5 >=  12
        8)   X3 - X2 >=  0
 END


     LP OPTIMUM FOUND  AT STEP     7

          OBJECTIVE FUNCTION VALUE

 1)          38.0000000

 VARIABLE         VALUE          REDUCED COST
        X6        38.000000          0.000000
        X1         0.000000          0.000000
        X3         9.000000          0.000000
        X2         9.000000          0.000000
        X5        26.000000          0.000000
        X4        16.000000          0.000000

    ROW      SLACK OR SURPLUS     DUAL PRICES
        2)         3.000000          0.000000
        3)         0.000000         -1.000000
        4)         9.000000          0.000000
        5)         0.000000         -1.000000
        6)         0.000000         -1.000000
        7)         0.000000         -1.000000
        8)         0.000000         -1.000000

 NO. ITERATIONS=       7


    RANGES IN WHICH THE BASIS IS UNCHANGED

                          OBJ COEFFICIENT RANGES
 VARIABLE         CURRENT        ALLOWABLE         ALLOWABLE
                   COEF          INCREASE          DECREASE
        X6         1.000000       INFINITY            0.000000
        X1        -1.000000       INFINITY            0.000000
        X3         1.000000       INFINITY            0.000000
        X2         1.000000       INFINITY            0.000000
        X5         1.000000       INFINITY            0.000000
        X4         1.000000       INFINITY            0.000000

                          RIGHTHAND SIDE RANGES
    ROW           CURRENT        ALLOWABLE         ALLOWABLE
                    RHS          INCREASE          DECREASE
         2         6.000000         3.000000        INFINITY
         3         9.000000       INFINITY            3.000000
         4         8.000000         9.000000        INFINITY
         5         7.000000       INFINITY            9.000000
         6        10.000000       INFINITY            9.000000
         7        12.000000       INFINITY           38.000000
         8         0.000000       INFINITY            3.000000
```

FIGURE 34
LINDO Output for Widgetco

TABLE 14

A	B	C	D	E	F
\$10	\$20	\$3	\$30	\$40	\$50

1091.ch08 5/13/03 1:23 PM Page 441

$$
\begin{aligned}
x_2 &\geq x_1 + 9 - B && \text{(Arc (1, 2) constraint)}\\
x_3 &\geq x_1 + 6 - A && \text{(Arc (1, 3) constraint)}\\
x_5 &\geq x_3 + 8 - C && \text{(Arc (3, 5) constraint)}\\
x_4 &\geq x_3 + 7 - D && \text{(Arc (3, 4) constraint)}\\
x_5 &\geq x_4 + 10 - E && \text{(Arc (4, 5) constraint)}\\
x_6 &\geq x_5 + 12 - F && \text{(Arc (5, 6) constraint)}\\
x_3 &\geq x_2 + 0 && \text{(Arc (2, 3) constraint)}\\
x_6 - x_1 &\leq 25 &&\\
&A, B, C, D, E, F \geq 0, x_j \text{urs} &&
\end{aligned}
$$

The first six constraints stipulate that the duration of each activity can be reduced by at most 5 days. As before, the next seven constraints ensure that event j cannot occur until after node i occurs and activity (i, j) is completed. For example, activity B (arc (1, 2)) now has a duration of $9 - B$. Thus, we need the constraint $x_2 \geq x_1 + (9 - B)$. The constraint $x_6 - x_1 \leq 25$ ensures that the project is completed within the 25-day deadline. The objective function is the total cost incurred in reducing the duration of the activities. An optimal solution to this LP is $z = \$390$, $x_1 = 0$, $x_2 = 4$, $x_3 = 4$, $x_4 = 6$, $x_5 = 13$, $x_6 = 25$, $A = 2$, $B = 5$, $C = 0$, $D = 5$, $E = 3$, $F = 0$. After reducing the durations of projects B, A, D, and E by the given amounts, we obtain the project network pictured in Figure 35. The reader should verify that A, B, D, E, and F are critical activities and that 1–2–3–4–5–6 and 1–3–4–5–6 are both critical paths (each having length 25). Thus, the project deadline of 25 days can be met for a cost of \$390.

Using LINGO to Determine the Critical Path

Many computer packages (such as Microsoft Project) enable the user to determine (among other things!) the critical path(s) and critical activities in a project network. You can always find a critical path and critical activities using LINDO, but LINGO makes it very easy to communicate the necessary information to the computer. The following LINGO program (file Widget1.lng) generates the objective function and constraints needed to find the critical path for the project network of Example 6 via linear programming.

Widget1.lng

```
MODEL:
  1]SETS:
  2]NODES/1..6/:TIME;
  3]ARCS(NODES,NODES)/
  4]1,2   1,3   2,3   3,4   3,5   4,5    5,6/:DUR;
  5]ENDSETS
  6]MIN=TIME(6)-TIME(1);
  7]@FOR(ARCS(I,J):TIME(J)>TIME(I)+DUR(I,J));
  8]DATA:
  9]DUR=9,6,0,7,8,10,12;
 10]ENDDATA
END
```

Line 1 begins the SETS portion of the program. In line 2, we define the six nodes of the project network and associate with each node a time that the events corresponding to

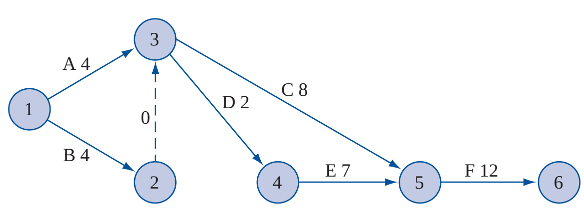


FIGURE 35
Duration of Activities after Crashing

1091.ch08 5/13/03 1:23 PM Page 442

```
MIN     -ET(1 + ET(6
SUBJECT TO
2)- ET(1 + ET(2 >=  9
3)- ET(1 + ET(3 >=  6
4)- ET(2 + ET(3 >=  0
5)- ET(3 + ET(4 >=  7
6)- ET(3 + ET(5 >=  8
7)- ET(4 + ET(5 >=  10
8)- ET(5 + ET(6 >=  12
END

LP OPTIMUM FOUND AT STEP      6
OBJECTIVE VALUE =    38.0000000

                     VARIABLE          VALUE          REDUCED COST
                       ET( 1)       0.0000000E+00       0.0000000E+00
                       ET( 2)        9.000000           0.0000000E+00
                       ET( 3)        9.000000           0.0000000E+00
                       ET( 4)        16.00000           0.0000000E+00
                       ET( 5)        26.00000           0.0000000E+00
                       ET( 6)        38.00000           0.0000000E+00
                   DUR( 1, 2)        9.000000           0.0000000E+00
                   DUR( 1, 3)        6.000000           0.0000000E+00
                   DUR( 2, 3)       0.0000000E+00       0.0000000E+00
                   DUR( 3, 4)        7.000000           0.0000000E+00
                   DUR( 3, 5)        8.000000           0.0000000E+00
                   DUR( 4, 5)        10.00000           0.0000000E+00
                   DUR( 5, 6)        12.00000           0.0000000E+00

                          ROW    SLACK OR SURPLUS     DUAL PRICE
                            1        38.00000           1.000000
                            2       0.0000000E+00      -1.000000
                            3        3.000000          0.0000000E+00
                            4       0.0000000E+00      -1.000000
                            5       0.0000000E+00      -1.000000
                            6        9.000000          0.0000000E+00
                            7       0.0000000E+00      -1.000000
                            8       0.0000000E+00      -1.000000
```

FIGURE 36

the node occurs. For example, TIME(3) represents the time when activities A and B have just been completed. In line 3, we generate the arcs in the project network by listing them (separated by spaces). For example, arc (3, 4) represents activity D. In line 4, we associate a duration (DUR) of each activity with each arc. Line 5 ends the SETS section of the program.

Line 6 specifies the objective, to minimize the time it takes to complete the project. For each arc defined in line 3, line 7 creates a constraint analagous to $x_j \geq x_i + t_{ij}$.

Line 8 begins the DATA section of the program. In line 9, we list the duration of each activity. Line 10 concludes the data entry and the **END** statement concludes the program. The output from this LINGO model is given in Figure 36, where by following the arcs corresponding to constraints having dual prices of -1, we find the critical path to be 1–2–3–4–5–6.

To find the critical path in any network we would begin by listing the nodes, arcs, and activity durations in our program. Then we would modify the objective function created by line 6 to reflect the number of nodes in the network. For example, if there were 10 nodes in the project network, we would change line 6 to **MIN**=TIME(10)–TIME(1); and we would be ready to go!

Widget2.lng

The following LINGO program (file Widget2.lng) enables the user to determine the critical path and total float at each node for Example 6 without using linear programming.

```
MODEL:
  1]MODEL:
  2]SETS:
  3]NODES/1..6/:ET,LT;
  4]ARCS(NODES,NODES)/1,2   1,3   2,3   3,4   3,5   4,5   5,6/:DUR,TFLOAT;
  5]ENDSETS
  6]DATA:
  7]DUR = 9,6,0,7,8,10,12;
```

```
  8]ENDDATA
  9]ET(1)=0;
 10]@FOR(NODES(J) | J#GT#1:
 11]ET(J) = @MAX(ARCS(I,J): ET(I)+DUR(I,J)););
 12]LNODE=@SIZE(NODES);
 13]LT(LNODE) = ET(LNODE);
 14]@FOR(NODES(I) | I#LT#LNODE:
 15]LT(I) = @MIN(ARCS(I,J): LT(J) - DUR(I,J)););
 16]@FOR(ARCS(I,J):TFLOAT(I,J)=LT(J)-ET(I)-DUR(I,J));
END
```

In line 3, we define the nodes of the project network and associate an early event time (ET) and late event time (LT) with each node. We define the arcs of the project network by listing them in line 4. With each arc we associate the duration of the arc's activity and the total float of the activity. In line 7, we input the duration of each activity.

To begin the computation of the ET(J)'s for each node, we set ET(1) = 0 in line 9. In lines 10–11, we compute ET(J) for all other nodes. For J > 1 ET(J) is the maximum value of ET(I) + DUR(I, J) for all (I, J) such that (I, J) is an arc in the network. By using the **@SIZE** function, which returns the number of elements in a set, we identify the finish node in the network in line 12. Thus, line 12 defines node 6 as the last node. In line 13, we set LT(6) = ET(6). Lines 14–15 work backward from node 6 toward node 1 to compute the LT(I)'s. For every node I other than the last node (6), LT(I) is the minimum of LT(J) − DUR (I, J), where the minimum is taken over all (I, J) such that (I, J) is an arc in the project network.

Finally, line 16 computes the total float for each activity (I, J) from total float for activity (I, J) = LT(Node J) − ET(Node I) − Duration (I, J). All activities whose total float equals 0 are critical activities.

After inputting a list of nodes, arcs, and activity durations we can use this program to analyze any project network (without changing any of lines 9–16). It is also easy to write a LINGO program that can be used to crash the network (see Problem 14).

PERT: Program Evaluation and Review Technique

CPM assumes that the duration of each activity is known with certainty. For many projects, this is clearly not applicable. PERT is an attempt to correct this shortcoming of CPM by modeling the duration of each activity as a random variable. For each activity, PERT requires that the project manager estimate the following three quantities:

a = estimate of the activity's duration under the most favorable conditions

b = estimate of the activity's duration under the least favorable conditions

m = most likely value for the activity's duration

Let $\mathbf{T}_{ij}$ (random variables are printed in boldface) be the duration of activity (i, j). PERT requires the assumption that $\mathbf{T}_{ij}$ follows a beta distribution. The specific definition of a beta distribution need not concern us, but it is important to realize that it can approximate a wide range of random variables, including many positively skewed, negatively skewed, and symmetric random variables. If $\mathbf{T}_{ij}$ follows a beta distribution, then it can be shown that the mean and variance of $\mathbf{T}_{ij}$ may be approximated by

$$E(\mathbf{T}_{ij}) = \frac{a + 4m + b}{6} \tag{4}$$

$$\mathrm{var}\mathbf{T}_{ij} = \frac{(b - a)^2}{36} \tag{5}$$

PERT requires the assumption that the durations of all activities are independent. Then for any path in the project network, the mean and variance of the time required to complete the activities on the path are given by

$$\sum_{(i,\ j)\in \text{path}} E(\mathbf{T}_{ij}) = \text{expected duration of activities on any path} \tag{6}$$

$$\sum_{(i,\ j)\in \text{path}} \text{var}\mathbf{T}_{ij} = \text{variance of duration of activities on any path} \tag{7}$$

Let **CP** be the random variable denoting the total duration of the activities on a critical path found by CPM. PERT assumes that the critical path found by CPM contains enough activities to allow us to invoke the Central Limit Theorem and conclude that

$$\mathbf{CP} = \sum_{(i,\ j)\in \text{critical path}} \mathbf{T}_{ij}$$

is normally distributed. With this assumption, (4)–(7) can be used to answer questions concerning the probability that the project will be completed by a given date. For example, suppose that for Example 6, a, b, and m for each activity are shown in Table 15. Now (4) and (5) yield

$$E(\mathbf{T}_{12}) = \frac{\{5 + 13 + 36\}}{6} = 9 \qquad \text{var}\mathbf{T}_{12} = \frac{(13 - 5)^2}{36} = 1.78$$

$$E(\mathbf{T}_{13}) = \frac{\{2 + 10 + 24\}}{6} = 6 \qquad \text{var}\mathbf{T}_{13} = \frac{(10 - 2)^2}{36} = 1.78$$

$$E(\mathbf{T}_{35}) = \frac{\{3 + 13 + 32\}}{6} = 8 \qquad \text{var}\mathbf{T}_{35} = \frac{(13 - 3)^2}{36} = 2.78$$

$$E(\mathbf{T}_{34}) = \frac{\{1 + 13 + 28\}}{6} = 7 \qquad \text{var}\mathbf{T}_{34} = \frac{(13 - 1)^2}{36} = 4$$

$$E(\mathbf{T}_{45}) = \frac{\{8 + 12 + 40\}}{6} = 10 \qquad \text{var}\mathbf{T}_{45} = \frac{(12 - 8)^2}{36} = 0.44$$

$$E(\mathbf{T}_{56}) = \frac{\{9 + 15 + 48\}}{6} = 12 \qquad \text{var}\mathbf{T}_{56} = \frac{(15 - 9)^2}{36} = 1$$

Of course, the fact that arc (2, 3) is a dummy arc yields

$$E(\mathbf{T}_{23}) = \text{var } \mathbf{T}_{23} = 0$$

Recall that the critical path for Example 6 was 1–2–3–4–5–6. From Equations (6) and (7),

$$E(\mathbf{CP}) = 9 + 0 + 7 + 10 + 12 = 38$$
$$\text{var } \mathbf{CP} = 1.78 + 0 + 4 + 0.44 + 1 = 7.22$$

Then the standard deviation for **CP** is $(7.22)^{1/2} = 2.69$.

TABLE 15
***a*, *b*, and *m* for Activities in Widgeto**

Activity	*a*	*b*	*m*
(1, 2)	5	13	9
(1, 3)	2	10	6
(3, 5)	3	13	8
(3, 4)	1	13	7
(4, 5)	8	12	10
(5, 6)	9	15	12

Applying the assumption that **CP** is normally distributed, we can answer questions such as the following: What is the probability that the project will be completed within 35 days? To answer this question, we must also make the following assumption: *No matter what the durations of the project's activities turn out to be, 1–2–3–4–5–6 will be a critical path.* This assumption implies that the probability that the project will be completed within 35 days is just $P(\mathbf{CP} \le 35)$. Standardizing and applying the assumption that **CP** is normally distributed, we find that **Z** is a standardized normal random variable with mean 0 and variance 1. The cumulative distribution function for a normal random variable is tabulated in Table 16. For example, $P(\mathbf{Z} \le -1) = 0.1587$ and $P(\mathbf{Z} \le 2) = 0.9772$. Thus,

$$P(\mathbf{CP} \le 35) = P\left(\frac{\mathbf{CP} - 38}{2.69} \le \frac{35 - 38}{2.69}\right) = P(\mathbf{Z} \le -1.12) = .13$$

where $F(-1.12) = .13$ may be obtained using the NORMSDIST function in Excel. Entering the formula =NORMSDIST(x) returns the probability that a standard normal random variable with mean 0 and standard deviation 1 is less than or equal to x. For example =NORMDIST(−1.12) yields .1313.

Difficulties with PERT

There are several difficulties with PERT:

1 The assumption that the activity durations are independent is difficult to justify.

2 Activity durations may not follow a beta distribution.

3 The assumption that the critical path found by CPM will always be the critical path for the project may not be justified.

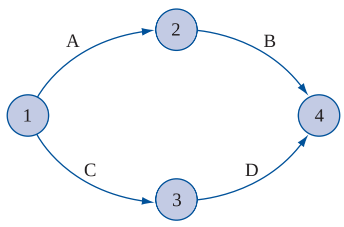
The last difficulty is the most serious. For example, in our analysis of Example 6, we assumed that 1–2–3–4–5–6 would always be the critical path. If, however, activity A were significantly delayed and activity B were completed ahead of schedule, then the critical path might be 1–3–4–5–6.

Here is a more concrete example of the fact that (because of the uncertain duration of activities) the critical path found by CPM may not actually be the path that determines the completion date of the project. Consider the simple project network in Figure 37. As-

TABLE 16
***a*, *b*, and *m* for Figure 37**

Activity	*a*	*b*	*m*
A	1	9	5
B	6	14	10
C	5	7	6
D	7	9	8

FIGURE 37
Project Network to Illustrate Difficulties with PERT



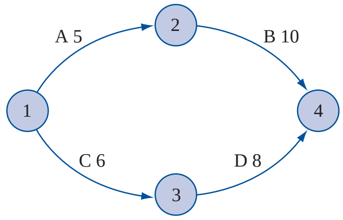


FIGURE 38
Network to Determine Critical Path If Each Activity's Duration Equals *m*

TABLE 17
Probability That Each Arc Is on a Critical Path

Activity	Probability
A	$\frac{17}{27}$
B	$\frac{17}{27}$
C	$\frac{12}{27}$
D	$\frac{12}{27}$

sume that for each activity in Table 16, *a*, *b*, and *m* each occur with probability $\frac{1}{3}$. If CPM were applied (using the expected duration of each activity as the duration of the activity), then we would obtain the network in Figure 38. For this network, the critical path is 1–2–4. In actuality, however, the critical path could be 1–3–4. For example, if the optimistic duration of B (6 days) occurred and all other activities had a duration *m*, then 1–3–4 would be the critical path in the network. If we assume that the durations of the four activities are independent random variables, then using elementary probability (see Problem 11 at the end of this section), it can be shown that there is a $\frac{10}{27}$ probability that 1–3–4 is the critical path, a $\frac{15}{27}$ chance that 1–2–4 is the critical path, and a $\frac{2}{27}$ chance that 1–2–4 and 1–3–4 will both be critical paths. This example shows that one must be cautious in designating an activity as critical. In this situation, the probability that each activity is actually a critical activity is shown in Table 17.

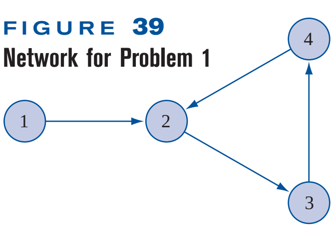
When the duration of activities is uncertain, the best way to analyze a project is to use a Monte Carlo simulation add-in for Excel. In Chapter 23, we will show how to use the Excel add-in @Risk to perform Monte Carlo simulations. With @Risk, we can easily determine the probability that a project is completed on time and determine the probability that each activity is critical.

PROBLEMS

Group A

1 What problem would arise if the network in Figure 39 were a portion of a project network?

FIGURE 39
Network for Problem 1



2 A company is planning to manufacture a product that consists of three parts (A, B, and C). The company anticipates that it will take 5 weeks to design the three parts and to determine the way in which these parts must be assembled to make the final product. Then the company estimates that it will take 4 weeks to make part A, 5 weeks to make part B, and 3 weeks to make part C. The company must test part A after it is completed (this takes 2 weeks). The assembly line process will then proceed as follows: assemble parts A and B (2 weeks) and then attach part C (1 week). Then the final product must undergo 1 week of

1091.ch08 5/13/03 1:23 PM Page 447

testing. Draw the project network and find the critical path, total float, and free float for each activity. Also set up the LP that could be used to find the critical path.

When determining the critical path in Problems 3 and 4, assume that m = activity duration.

3 Consider the project network in Figure 40. For each activity, you are given the estimates of a, b, and m in Table 18. Determine the critical path for this network, the total float for each activity, the free float for each activity, and the probability that the project is completed within 40 days. Also set up the LP that could be used to find the critical path.

4 The promoter of a rock concert in Indianapolis must perform the tasks shown in Table 19 before the concert can be held (all durations are in days).

a Draw the project network.

b Determine the critical path.

c If the advance promoter wants to have a 99% chance of completing all preparations by June 30, when should work begin on finding a concert site?

d Set up the LP that could be used to find the project's critical path.

5 Consider the (simplified) list of activities and predecessors that are involved in building a house (Table 20).

FIGURE 40
Network for Problem 3

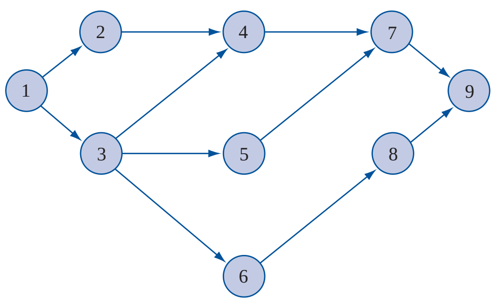


TABLE 18

Activity	a	b	m
(1, 2)	4	8	6
(1, 3)	2	8	4
(2, 4)	1	7	3
(3, 4)	6	12	9
(3, 5)	5	15	10
(3, 6)	7	18	12
(4, 7)	5	12	9
(5, 7)	1	3	2
(6, 8)	2	6	3
(7, 9)	10	20	15
(8, 9)	6	11	9

TABLE 19

Activity	Description	Immediate Predecessors	a	b	m
A	Find site	—	2	4	3
B	Find engineers	A	1	3	2
C	Hire opening act	A	2	10	6
D	Set radio and TV ads	C	1	3	2
E	Set up ticket agents	A	1	5	3
F	Prepare electronics	B	2	4	3
G	Print advertising	C	3	7	5
H	Set up transportation	C	0.5	1.5	1
I	Rehearsals	F, H	1	2	1.5
J	Last-minute details	I	1	3	2

TABLE 20

Activity	Description	Immediate Predecessors	Duration (Days)
A	Build foundation	—	5
B	Build walls and ceilings	A	8
C	Build roof	B	10
D	Do electrical wiring	B	5
E	Put in windows	B	4
F	Put on siding	E	6
G	Paint house	C, F	3

a Draw a project network, determine the critical path, find the total float for each activity, and find the free float for each activity.

b Suppose that by hiring additional workers, the duration of each activity can be reduced. The costs per day of reducing the duration of the activities are given in Table 21. Write down the LP to be solved to minimize the total cost of completing the project within 20 days.

6 Horizon Cable is about to expand its cable TV offerings in Smalltown by adding MTV and other exciting stations. The activities in Table 22 must be completed before the service expansion is completed.

a Draw the project network and determine the critical path for the network, the total float for each activity, and the free float for each activity.

b Set up the LP that can be used to find the project's critical path.

7 When an accounting firm audits a corporation, the first phase of the audit involves obtaining "knowledge of the business." This phase of the audit requires the activities in Table 23.

a Draw the project network and determine the critical path for the network, the total float for each activity, and the free float for each activity. Also set up the LP that can be used to find the project's critical path.

1091.ch08 5/13/03 1:23 PM Page 448

TABLE 21

Activity	Cost per Day of Reducing Duration of Activity ($)	Maximum Possible Reduction in Duration of Activity (Days)
Foundation	30	2
Walls and ceiling	15	3
Roof	20	1
Electrical wiring	40	2
Windows	20	2
Siding	30	3
Paint	40	1

TABLE 22

Activity	Description	Immediate Predecessors	Duration (Weeks)
A	Choose stations	—	2
B	Get town council to approve expansion	A	4
C	Order converters needed to expand service	B	3
D	Install new dish to receive new stations	B	2
E	Install converters	C, D	10
F	Change billing system	B	4

TABLE 23

Activity	Description	Immediate Predecessors	Duration (Days)
A	Determining terms of engagement	—	3
B	Appraisal of auditability risk and materiality	A	6
C	Identification of types of transactions and possible errors	A	14
D	Systems description	C	8
E	Verification of systems description	D	4
F	Evaluation of internal controls	B, E	8
G	Design of audit approach	F	9

b Assume that the project must be completed in 30 days. The duration of each activity can be reduced by incurring the costs shown in Table 24. Formulate an LP that can be used to minimize the cost of meeting the project deadline.

8 The LINDO output in Figure 41 can be used to determine the critical path for Problem 5. Use this output to do the following:

TABLE 24

Activity	Cost per Day of Reducing Duration of Activity ($)	Maximum Possible Reduction in Duration of Activity (Days)
A	100	3
B	80	4
C	60	5
D	70	2
E	30	4
F	20	4
G	50	4

FIGURE 41

LINDO Output for Problem 8

```
     MIN      X6 - X1
         SUBJECT TO
                2) - X1 + X2 >=  5
                3) - X2 + X3 >=  8
                4) - X3 + X4 >=  4
                5) - X3 + X5 >=  10
                6) - X4 + X5 >=  6
                7)   X6 - X3 >=  5
                8)   X6 - X5 >=  3
         END


            LP OPTIMUM FOUND  AT STEP     6

                 OBJECTIVE FUNCTION VALUE

          1)        26.0000000

         VARIABLE        VALUE          REDUCED COST
                X6       26.000000          0.000000
                X1        0.000000          0.000000
                X2        5.000000          0.000000
                X3       13.000000          0.000000
                X4       17.000000          0.000000
                X5       23.000000          0.000000

            ROW      SLACK OR SURPLUS     DUAL PRICES
                2)        0.000000         -1.000000
                3)        0.000000         -1.000000
                4)        0.000000         -1.000000
                5)        0.000000          0.000000
                6)        0.000000         -1.000000
                7)        8.000000          0.000000
                8)        0.000000         -1.000000

         NO. ITERATIONS=       6


   RANGES IN WHICH THE BASIS IS UNCHANGED

                                OBJ COEFFICIENT RANGES
VARIABLE         CURRENT        ALLOWABLE         ALLOWABLE
                  COEF          INCREASE          DECREASE
     X6          1.000000       INFINITY           0.000000
     X1         -1.000000       INFINITY           0.000000
     X2          0.000000       INFINITY           0.000000
     X3          0.000000       INFINITY           0.000000
     X4          0.000000       INFINITY           0.000000
     X5          0.000000       INFINITY           0.000000

                              RIGHTHAND SIDE RANGES
ROW            CURRENT         ALLOWABLE         ALLOWABLE
                 RHS           INCREASE          DECREASE
          2     5.000000        INFINITY           5.000000
          3     8.000000        INFINITY          13.000000
          4     4.000000          0.000000         8.000000
          5    10.000000        INFINITY           0.000000
          6     6.000000          0.000000         8.000000
          7     5.000000          8.000000       INFINITY
          8     3.000000        INFINITY           8.000000
```

1091.ch08 5/13/03 1:23 PM Page 449

a Draw the project diagram.

b Determine the length of the critical path and the critical activities for this project.

9 Explain why an activity's free float can never exceed the activity's total float.

10 A project is complete when activities A–E are completed. The predecessors of each activity are shown in Table 25. Draw the appropriate project diagram. (*Hint:* Don't violate rule 4.)

11 Determine the probabilities that 1–2–4 and 1–3–4 are critical paths for Figure 37.

12 Given the information in Table 26, **(a)** draw the appropriate project network, and **(b)** find the critical path.

13 The government is going to build a high-speed computer in Austin, Texas. Once the computer is designed (D), we can select the exact site (S), the building contractor (C), and the operating personnel (P). Once the site is selected, we can begin erecting the building (B). We can start manufacturing the computer (COM) and preparing the operations manual (M) only after contractor is selected. We can begin training the computer operators (T) when the operating manual and personnel selection are completed. When the computer and the building are both finished, the computer may be installed (I). Then the computer is considered operational. Draw a project network that could be used to determine when the project is operational.

14 Write a LINGO program that can be used to crash the project network of Example 6 with the crashing costs given in Table 14.

15 Consider the project diagram in Figure 42. This project must be completed in 90 days. The time required to complete each activity can be reduced by up to five days at the costs given in Table 27.

Formulate an LP whose solution will enable us to minimize the cost of completing the project in 90 days.

16–17 Find the critical path, total float, and free float for each activity in the project networks of Figures 43 and 44.

TABLE 25

Activity	Predecessors
A	—
B	A
C	A
D	B
E	B, C

TABLE 26

Activity	Immediate Predecessors	Duration (Days)
A	—	3
B	—	3
C	—	1
D	A, B	3
E	A, B	3
F	B, C	2
G	D, E	4
H	E	3

FIGURE 42

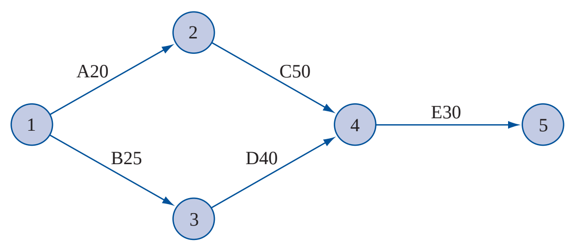


TABLE 27

Activity	Cost of Reducing Activities Duration by 1 Day ($)
A	300
B	200
C	350
D	260
E	320

FIGURE 43

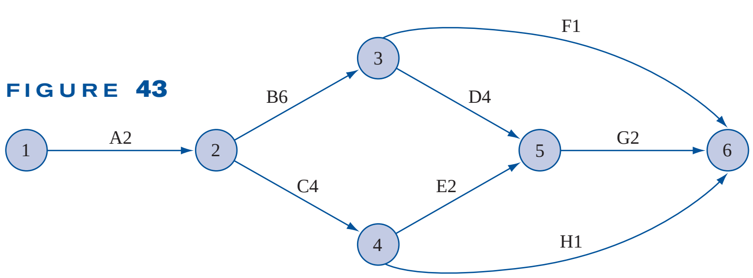
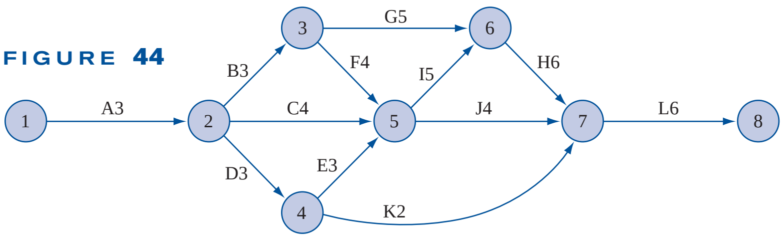


FIGURE 44



8.5 Minimum-Cost Network Flow Problems

The transportation, assignment, transshipment, shortest-path, maximum flow, and CPM problems are all special cases of the minimum-cost network flow problem (MCNFP). Any MCNFP can be solved by a generalization of the transportation simplex called the **network simplex.**

To define an MCNFP, let

x_{ij} = number of units of flow sent from node i to node j through arc (i, j)

b_i = net supply (outflow − inflow) at node i

c_{ij} = cost of transporting 1 unit of flow from node i to node j via arc (i, j)

L_{ij} = lower bound on flow through arc (i, j) (if there is no lower bound, let $L_{ij} = 0$)

U_{ij} = upper bound on flow through arc (i, j) (if there is no upper bound, let $U_{ij} = \infty$)

Then the MCNFP may be written as

$$\min \sum_{\text{all arcs}} c_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{ij} - \sum_k x_{ki} = b_i \qquad \text{(for each node } i \text{ in the network)} \tag{8}$$

$$L_{ij} \le x_{ij} \le U_{ij} \qquad \text{(for each arc in the network)} \tag{9}$$

Constraints (8) stipulate that the net flow out of node i must equal b_i. Constraints (8) are referred to as the **flow balance equations** for the network. Constraints (9) ensure that the flow through each arc satisfies the arc capacity restrictions. In all our previous examples, we have set $L_{ij} = 0$.

Let us show that transportation and maximum-flow problems are special cases of the minimum-cost network flow problem.

Formulating a Transportation Problem as an MCNFP

Consider the transportation problem in Table 28. Nodes 1 and 2 are the two supply points, and nodes 3 and 4 are the two demand points. Then $b_1 = 4$, $b_2 = 5$, $b_3 = -6$, and $b_4 = -3$. The network corresponding to this transportation problem contains arcs (1, 3), (1, 4), (2, 3), and (2, 4) (see Figure 45). The LP for this transportation problem may be written as shown in Table 29.

The first two constraints are the supply constraints, and the last two constraints are (after being multiplied by −1) the demand constraints. Because this transportation problem

1091.ch08 6/22/06 2:34 PM Page 451

TABLE 28

	1	2	4 (Node 1)
	3	4	5 (Node 2)
	6 (Node 3)	3 (Node 4)	

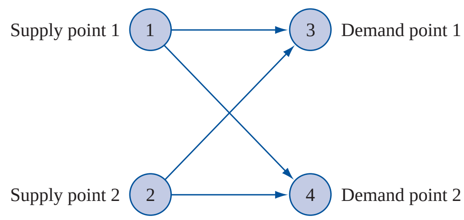


FIGURE 45
Representation of Transportation Problem as an MCNFP

TABLE 29
MCNFP Representation of Transportation Problem

x_{13}	x_{14}	x_{23}	x_{24}		rhs	Constraint
			$\min z = x_{13} + 2x_{14} + 3x_{23} + 4x_{24}$			
1	1	0	0	=	4	Node 1
0	0	1	1	=	5	Node 2
−1	0	−1	0	=	−6	Node 3
0	−1	0	−1	=	−3	Node 4
			All variables non-negative			

has no arc capacity restrictions, the flow balance equations are the only constraints. We note that if the problem had not been balanced, we could not have formulated the problem as an MCNFP. This is because if total supply exceeded total demand, we would not know with certainty the net outflow at each supply point. Thus, to formulate a transportation (or a transshipment) problem as an MCNFP, it may be necessary to add a dummy point.

Formulating a Maximum-Flow Problem as an MCNFP

To see how a maximum-flow problem fits into the minimum-cost network flow context, consider the problem of finding the maximum flow from source to sink in the network of Figure 6. After creating an arc a_0 joining the sink to the source, we have $b_{so} = b_1 = b_2 = b_3 = b_{si} = 0$. Then the LP constraints for finding the maximum flow in Figure 6 may be written as shown in Table 30.

The first five constraints are the flow balance equations for the nodes of the network, and the last six constraints are the arc capacity constraints. Because there is no upper limit on the flow through the artificial arc, there is no arc capacity constraint for a_0.

The flow balance equations in any MCNFP have the following important property: *Each variable x_{ij} has a coefficient of $+1$ in the node i flow balance equation, a coefficient of -1 in the node j flow balance equation, and a coefficient of 0 in all other flow balance equations.* For example, in a transportation problem, the variable x_{ij} will have a coeffi-

TABLE 30
MCNFP Representation of Maximum-Flow Problem

min $z = x_0$									
$x_{so,1}$	$x_{so,2}$	x_{13}	x_{12}	$x_{3,si}$	$x_{2,si}$	x_0		rhs	Constraint
1	1	0	0	0	0	−1	=	0	Node *so*
−1	0	1	1	0	0	0	=	0	Node 1
0	−1	0	−1	0	1	0	=	0	Node 2
0	0	−1	0	1	0	0	=	0	Node 3
0	0	0	0	−1	−1	1	=	0	Node *si*
1	0	0	0	0	0	0	≤	2	Arc (*so*, 1)
0	1	0	0	0	0	0	≤	3	Arc (*so*, 2)
0	0	1	0	0	0	0	≤	4	Arc (1, 3)
0	0	0	1	0	0	0	≤	3	Arc (1, 2)
0	0	0	0	1	0	0	≤	1	Arc (3, *si*)
0	0	0	0	0	1	0	≤	2	Arc (2, *si*)
All variables nonnegative									

cient of +1 in the flow balance equation for supply point i, a coefficient of −1 in the flow balance equation for demand point j, and a coefficient of 0 in all other flow balance equations. Even if the constraints of an LP do not appear to contain the flow balance equations of a network, clever transformation of an LP's constraints can often show that an LP is equivalent to an MCNFP (see Problem 6 at the end of this section).

An MCNFP can be solved by a generalization of the transportation simplex known as the *network simplex algorithm* (see Section 8.7). As with the transportation simplex, the pivots in the network simplex involve only additions and subtractions. This fact can be used to prove that if all the b_i's and arc capacities are integers, then in the optimal solution to an MCNFP, all the variables will be integers. Computer codes that use the network simplex can quickly solve even extremely large network problems. For example, MCNFPs with 5,000 nodes and 600,000 arcs have been solved in under 10 minutes. To use a network simplex computer code, the user need only input a list of the network's nodes and arcs, the c_{ij}'s and arc capacity for each arc, and the b_i's for each node. The network simplex is efficient and easy to use, so it is extremely important to formulate an LP, if at all possible, as an MCNFP.

To close this section, we formulate a simple traffic assignment problem as an MCNFP.

EXAMPLE 7 Traffic MCNFP

Each hour, an average of 900 cars enter the network in Figure 46 at node 1 and seek to travel to node 6. The time it takes a car to traverse each arc is shown in Table 31. In Figure 46, the number above each arc is the maximum number of cars that can pass by any point on the arc during a one-hour period. Formulate an MCNFP that minimizes the total time required for all cars to travel from node 1 to node 6.

Solution Let

$$x_{ij} = \text{number of cars per hour that traverse the arc from node } i \text{ to node } j$$

Then we want to minimize

$$z = 10x_{12} + 50x_{13} + 70x_{25} + 30x_{24} + 30x_{56} + 30x_{45} + 60x_{46} + 60x_{35} + 10x_{34}$$

We are given that $b_1 = 900$, $b_2 = b_3 = b_4 = b_5 = 0$, and $b_6 = -900$ (we will not introduce the artificial arc connecting node 6 to node 1). The constraints for this MCNFP are shown in Table 32.

1091.ch08 5/13/03 1:23 PM Page 453

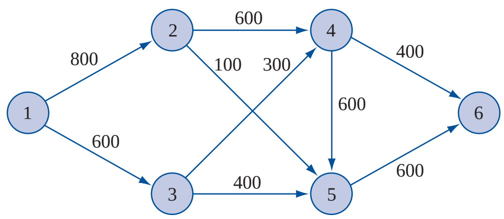


FIGURE 46
Representation of Traffic Example as MCNFP

TABLE 31
Travel Times for Traffic Example

Arc	Time (Minutes)
(1, 2)	10
(1, 3)	50
(2, 5)	70
(2, 4)	30
(5, 6)	30
(4, 5)	30
(4, 6)	60
(3, 5)	60
(3, 4)	10

TABLE 32
MCNFP Representation of Traffic Example

x_{12}	x_{13}	x_{24}	x_{25}	x_{34}	x_{35}	x_{45}	x_{46}	x_{56}		rhs	Constraint
1	1	0	0	0	0	0	0	0	=	900	Node 1
−1	0	1	1	0	0	0	0	0	=	0	Node 2
0	−1	0	0	1	1	0	0	0	=	0	Node 3
0	0	−1	0	−1	0	1	1	0	=	0	Node 4
0	0	0	−1	0	−1	−1	0	1	=	0	Node 5
0	0	0	0	0	0	0	−1	−1	=	−900	Node 6
1	0	0	0	0	0	0	0	0	≤	800	Arc (1, 2)
0	1	0	0	0	0	0	0	0	≤	600	Arc (1, 3)
0	0	1	0	0	0	0	0	0	≤	600	Arc (2, 4)
0	0	0	1	0	0	0	0	0	≤	100	Arc (2, 5)
0	0	0	0	1	0	0	0	0	≤	300	Arc (3, 4)
0	0	0	0	0	1	0	0	0	≤	400	Arc (3, 5)
0	0	0	0	0	0	1	0	0	≤	600	Arc (4, 5)
0	0	0	0	0	0	0	1	0	≤	400	Arc (4, 6)
0	0	0	0	0	0	0	0	1	≤	600	Arc (5, 6)
All variables non-negative											

Solving an MCNFP with LINGO

Traffic.lng

The following LINGO program (file Traffic.lng) can be used to find the optimal solution to Example 7 (or any MCNFP).

1091.ch08 5/13/03 1:23 PM Page 454

```
MODEL:
  1] SETS:
  2] NODES/1..6/:SUPP;
  3] ARCS(NODES,NODES)/1,2  1,3  2,4  2,5  3,4  3,5  4,5  4,6  5,6/
  4] :CAP,FLOW,COST;
  5] ENDSETS
  6] MIN=@SUM(ARCS:COST*FLOW);
  7] @FOR(ARCS(I,J):FLOW(I,J)<CAP(I,J));
  8] @FOR(NODES(I):-@SUM(ARCS)(J,I):FLOW(J,I))
  9] +@SUM(ARCS(I,J):FLOW(I,J))=SUPP(I));
 10] DATA:
 11] COST=10,50,30,70,10,60,30,60,30;
 12] SUPP=900,0,0,0,0,-900;
 13] CAP=800,600,600,100,300,400,600,400,600;
 14] ENDDATA
END
```

In line 2, we define the network's nodes and associate a net supply (flow out−flow in) with each node. The supplies data are entered in line 12. In line 3, we define, by listing, the arcs in the network and in line 4 associate a capacity (CAP), a flow (FLOW), and a cost-per-unit-shipped (COST) with each arc. The unit shipping costs data are entered in line 11. Line 6 generates the objective function by summing over all arcs (unit cost for arc)*(flow through arc). Line 7 generates each arc's capacity constraint (arc capacities data are entered in line 13). For each node, lines 8–9 generate the conservation-of-flow constraint. They imply that for each node I, −(flow into node I) + (flow out of node I) = (supply of node I). When solved on LINGO, we find that the solution to Example 7 is z =95,000 minutes, $x_{12} = 700$, $x_{13} = 200$, $x_{24} = 600$, $x_{25} = 100$, $x_{34} = 200$, $x_{45} = 400$, $x_{46} = 400$, $x_{56} = 500$.

Our LINGO program can be used to solve any MCNFP. Just input the set of nodes, supplies, arcs, and unit shipping cost; hit **GO** and you are done!

PROBLEMS

Note: To formulate a problem as an MCNFP, you should draw the appropriate network and determine the c_{ij}'s, the b_i's, and the arc capacities.

Group A

1 Formulate the problem of finding the shortest path from node 1 to node 6 in Figure 2 as an MCNFP. (*Hint:* Think of finding the shortest path as the problem of minimizing the total cost of sending 1 unit of flow from node 1 to node 6.)

2 **a** Find the dual of the LP that was used to find the length of the critical path for Example 6 of Section 8.4.

b Show that the answer in part (a) is an MCNFP.

c Explain why the optimal objective function value for the LP found in part (a) is the longest path in the project network from node 1 to node 6. Why does this justify our earlier claim that the critical path in a project network is the longest path from the start node to the finish node?

3 Fordco produces cars in Detroit and Dallas. The Detroit plant can produce as many as 6,500 cars, and the Dallas plant can produce as many as 6,000 cars. Producing a car costs $2,000 in Detroit and $1,800 in Dallas. Cars must be shipped to three cities. City 1 must receive 5,000 cars, city 2 must receive 4,000 cars, and city 3 must receive 3,000 cars. The cost of shipping a car from each plant to each city is given in Table 33. At most, 2,200 cars may be sent from a given plant to a given city. Formulate an MCNFP that can be used to minimize the cost of meeting demand.

4 Each year, Data Corporal produces as many as 400 computers in Boston and 300 computers in Raleigh. Los Angeles customers must receive 400 computers, and 300 computers must be supplied to Austin customers. Producing a computer costs $800 in Boston and $900 in Raleigh. Computers are transported by plane and may be sent through Chicago. The costs of sending a computer between pairs of cities are shown in Table 34.

a Formulate an MCNFP that can be used to minimize the total (production + distribution) cost of meeting Data Corporal's annual demand.

TABLE 33

	To ($)		
From	City 1	City 2	City 3
Detroit	800	600	300
Dallas	500	200	200

TABLE 34

		To ($)	
From	Chicago	Austin	Los Angeles
Boston	80	220	280
Raleigh	100	140	170
Chicago	—	40	50

b How would you modify the part (a) formulation if at most 200 units could be shipped through Chicago? [*Hint:* Add an additional node and arc to this part (a) network.]

5 Oilco has oil fields in San Diego and Los Angeles. The San Diego field can produce 500,000 barrels per day, and the Los Angeles field can produce 400,000 barrels per day. Oil is sent from the fields to a refinery, in either Dallas or Houston (assume each refinery has unlimited capacity). To refine 100,000 barrels costs $700 at Dallas and $900 at Houston. Refined oil is shipped to customers in Chicago and New York. Chicago customers require 400,000 barrels per day, and New York customers require 300,000 barrels per day. The costs of shipping 100,000 barrels of oil (refined or unrefined) between cities are shown in Table 35.

a Formulate an MCNFP that can be used to determine how to minimize the total cost of meeting all demands.

b If each refinery had a capacity of 500,000 barrels per day, how would the part (a) answer be modified?

Group B

6 Workco must have the following number of workers available during the next three months: month 1, 20; month 2, 16; month 3, 25. At the beginning of month 1, Workco has no workers. It costs Workco $100 to hire a worker and $50 to fire a worker. Each worker is paid a salary of $140/month. We will show that the problem of determining a hiring and firing strategy that minimizes the total cost incurred during the next three (or in general, the next *n*) months can be formulated as an MCNFP.

a Let

x_{ij} = number of workers hired at beginning of month i and fired after working till end of month $j - 1$

(if $j = 4$, the worker is never fired). Explain why the following LP will yield a minimum-cost hiring and firing strategy:

TABLE 35

			To ($)	
From	Dallas	Houston	New York	Chicago
Los Angeles	300	110	—	—
San Diego	420	100	—	—
Dallas	—	—	450	550
Houston	—	—	470	530

$$
\begin{aligned}
\min z = {} & 50(x_{12} + x_{13} + x_{23}) \\
& + 100(x_{12} + x_{13} + x_{14} + x_{23} + x_{24} + x_{34}) \\
& + 140(x_{12} + x_{23} + x_{34}) \\
& + 280(x_{13} + x_{24}) + 420x_{14}
\end{aligned}
$$

$$
\begin{aligned}
\text{s.t.} \quad & (1) \quad x_{12} + x_{13} + x_{14} - e_1 = 20 && \text{(Month 1 constraint)} \\
& (2) \quad x_{13} + x_{14} + x_{23} + x_{24} - e_2 = 16 && \text{(Month 2 constraint)} \\
& (3) \quad x_{14} + x_{24} + x_{34} - e_3 = 25 && \text{(Month 3 constraint)} \\
& \qquad x_{ij} \geq 0
\end{aligned}
$$

b To obtain an MCNFP, replace the constraints in part (a) by

i Constraint (1);

ii Constraint (2) − Constraint (1);

iii Constraint (3) − Constraint (2);

iv − (Constraint (3)).

Explain why an LP with Constraints (i)–(iv) is an MCNFP.

c Draw the network corresponding to the MCNFP obtained in answering part (b).

7[†] Braneast Airlines must determine how many airplanes should serve the Boston–New York–Washington air corridor and which flights to fly. Braneast may fly any of the daily flights shown in Table 36. The fixed cost of operating an airplane is $800/day. Formulate an MCNFP that can be used to maximize Braneast's daily profits. (*Hint:* Each node in the network represents a city and a time. In addition to arcs representing flights, we must allow for the possibility that an airplane will stay put for an hour or more. We must ensure that the model includes the fixed cost of operating a plane. To include this cost, the following three arcs might be included in the network: from Boston 7 P.M. to Boston 9 A.M.; from New York 7 P.M. to New York 9 A.M.; and from Washington 7 P.M. to Washington 9 A.M.)

8 Daisymay Van Line moves people between New York, Philadelphia, and Washington, D.C. It takes a van one day to travel between any two of these cities. The company incurs costs of $1,000 per day for a van that is fully loaded and traveling, $800 per day for an empty van that travels, $700 per day for a fully loaded van that stays in a city, and $400 per day for an empty van that remains in a city. Each day of the week, the loads described in Table 37 must be shipped. On Monday, for example, two trucks must be sent from Philadelphia to New York (arriving on Tuesday). Also, two trucks must be sent from Philadelphia to Washington on Friday (assume that Friday shipments must arrive on Monday). Formulate an MCNFP that can be used to minimize the cost of meeting weekly requirements. To simplify the formulation, assume that the requirements repeat each week. Then it seems plausible to assume that any of the company's trucks will begin each week in the same city in which it began the previous week.

[†]This problem is based on Glover et al. (1982).

TABLE 36

Leaves		Arrives		Flight Revenue	Variable Cost of Flight ($)
City	Time	City	Time		
N.Y.	9 A.M.	Wash.	10 A.M.	$900	400
N.Y.	2 P.M.	Wash.	3 P.M.	$600	350
N.Y.	10 A.M.	Bos.	11 A.M.	$800	400
N.Y.	4 P.M.	Bos.	5 P.M.	$1,200	450
Wash.	9 A.M.	N.Y.	10 A.M.	$1,100	400
Wash.	3 P.M.	N.Y.	4 P.M.	$900	350
Wash.	10 A.M.	Bos.	12 noon	$1,500	700
Wash.	5 P.M.	Bos.	7 P.M.	$1,800	900
Bos.	10 A.M.	N.Y.	11 A.M.	$900	500
Bos.	2 P.M.	N.Y.	3 P.M.	$800	450
Bos.	11 A.M.	Wash.	1 P.M.	$1,100	600
Bos.	3 P.M.	Wash.	5 P.M.	$1,200	650

TABLE 37

Trip	Monday	Tuesday	Wednesday	Thursday	Friday
Phil.–N.Y.	2	—	—	—	—
Phil.–Wash.	—	2	—	—	2
N.Y.–Phil.	3	2	—	—	—
N.Y.–Wash.	—	—	2	2	—
N.Y.–Phil.	1	—	—	—	—
Wash.–N.Y.	—	—	1	—	1

8.6 Minimum Spanning Tree Problems

Suppose that each arc (i, j) in a network has a length associated with it and that arc (i, j) represents a way of connecting node i to node j. For example, if each node in a network represents a computer at State University, then arc (i, j) might represent an underground cable that connects computer i with computer j. In many applications, we want to determine the set of arcs in a network that connect all nodes such that the sum of the length of the arcs is minimized. Clearly, such a group of arcs should contain no loop. (A loop is often called a *closed path* or *cycle.*) For example, in Figure 47, the sequence of arcs (1, 2)–(2, 3)–(3, 1) is a loop.

DEFINITION ■ For a network with n nodes, a **spanning tree** is a group of $n - 1$ arcs that connects all nodes of the network and contains no loops. ■

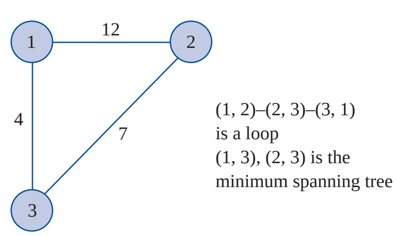


FIGURE 47
Illustration of Loop and Minimum Spanning Tree

In Figure 47, there are three spanning trees:

1 Arcs (1, 2) and (2, 3)

2 Arcs (1, 2) and (1, 3)

3 Arcs (1, 3) and (2, 3)

A spanning tree of minimum length in a network is a **minimum spanning tree (MST).** In Figure 47, the spanning tree consisting of arcs (1, 3) and (2, 3) is the unique minimum spanning tree.

The following method (MST algorithm) may be used to find a minimum spanning tree.

Step 1 Begin at any node i, and join node i to the node in the network (call it node j) that is closest to node i. The two nodes i and j now form a connected set of nodes $C = \{i, j\}$, and arc (i, j) will be in the minimum spanning tree. The remaining nodes in the network (call them C') are referred to as the *unconnected* set of nodes.

Step 2 Now choose a member of C' (call it n) that is closest to some node in C. Let m represent the node in C that is closest to n. Then the arc (m, n) will be in the minimum spanning tree. Now update C and C'. Because n is now connected to $\{i, j\}$, C now equals $\{i, j, n\}$ and we must eliminate node n from C'.

Step 3 Repeat this process until a minimum spanning tree is found. Ties for closest node and arc to be included in the minimum spanning tree may be broken arbitrarily.

At each step the algorithm chooses the shortest arc that can be used to expand C, so the algorithm is often referred to as a "greedy" algorithm. It is remarkable that the act of being "greedy" at each step of the algorithm can never force us later to follow a "bad arc." In Example 1 of Chapter 9 we will see that for some types of problems, a greedy algorithm may not yield an optimal solution! A justification of the MST algorithm is given in Problem 3 at the end of this section. Example 8 illustrates the algorithm.

EXAMPLE 8 MST Algorithm

The State University campus has five minicomputers. The distance between each pair of computers (in city blocks) is given in Figure 48. The computers must be interconnected by underground cable. What is the minimum length of cable required? Note that if no arc is drawn connecting a pair of nodes, this means that (because of underground rock formations) no cable can be laid between these two computers.

Solution We want to find the minimum spanning tree for Figure 48.

Iteration 1 Following the MST algorithm, we arbitrarily choose to begin at node 1. The closest node to node 1 is node 2. Now $C = \{1, 2\}$, $C' = \{3, 4, 5\}$, and arc (1, 2) will be in the minimum spanning tree (see Figure 49a).

Iteration 2 Node 5 is closest (two blocks distant) to C. Because node 5 is two blocks from node 1 and from node 2, we may include either arc (2, 5) or arc (1, 5) in the minimum spanning tree. We arbitrarily choose to include arc (2, 5). Then $C = \{1, 2, 5\}$ and $C' = \{3, 4\}$ (see Figure 49b).

Iteration 3 Node 3 is two blocks from node 5, so we may include arc (5, 3) in the minimum spanning tree. Now $C = \{1, 2, 3, 5\}$ and $C' = 4$ (see Figure 49c).

Iteration 4 Node 5 is the closest node to node 4, so we add arc (5, 4) to the minimum spanning tree (see Figure 49d).

We have now obtained the minimum spanning tree consisting of arcs (1, 2), (2, 5), (5, 3), and (5, 4). The length of the minimum spanning tree is $1 + 2 + 2 + 4 = 9$ blocks.

1091.ch08 5/13/03 1:23 PM Page 458

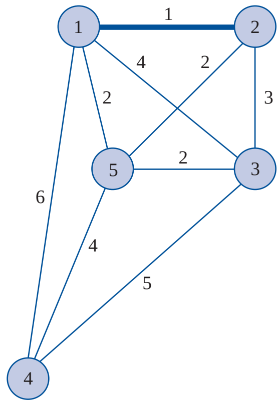


FIGURE 48
Distances between State University Computers

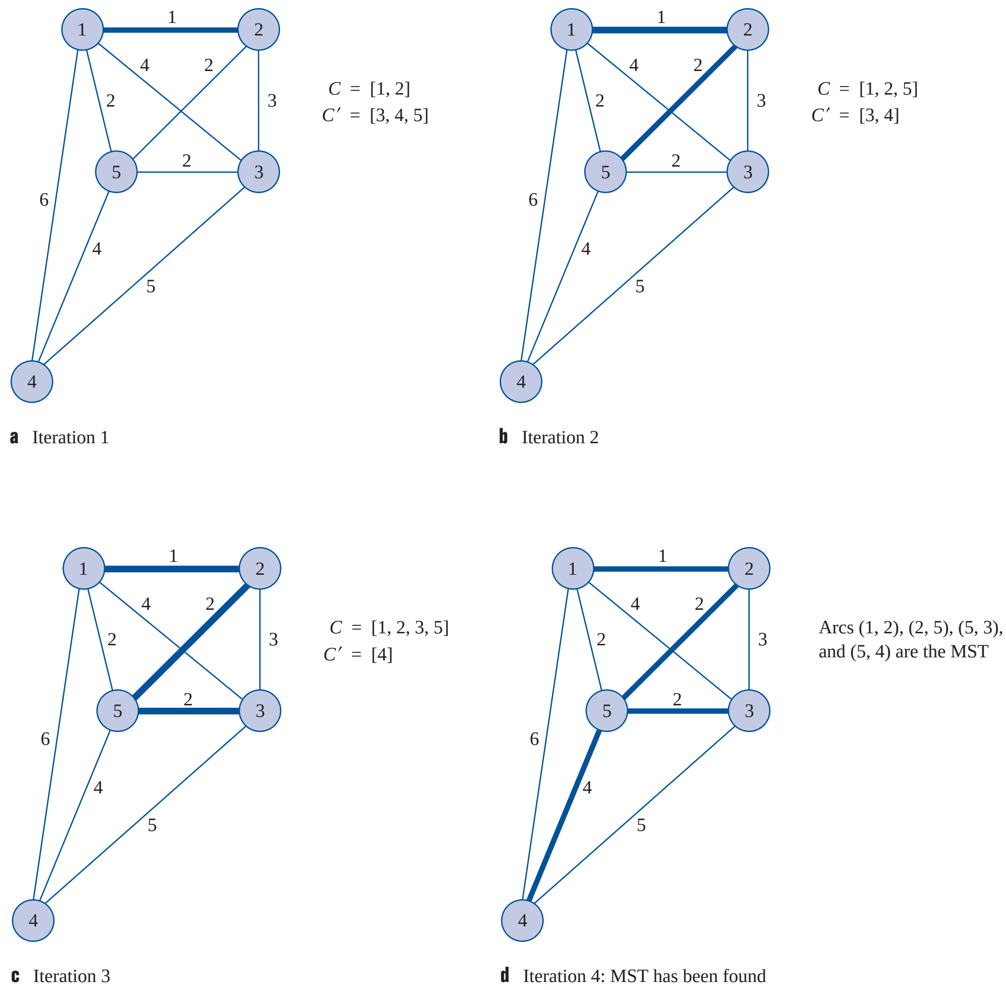


FIGURE 49
MST Algorithm for Computer Example

1091.ch08 5/13/03 1:23 PM Page 459

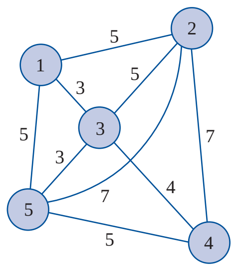
PROBLEMS

Group A

1 The distances (in miles) between the Indiana cities of Gary, Fort Wayne, Evansville, Terre Haute, and South Bend are shown in Table 38. It is necessary to build a state road system that connects all these cities. Assume that for political reasons no road can be built connecting Gary and Fort Wayne, and no road can be built connecting South Bend and Evansville. What is the minimum length of road required?

2 The city of Smalltown consists of five subdivisions. Mayor John Lion wants to build telephone lines to ensure that all the subdivisions can communicate with each other. The distances between the subdivisions are given in Figure 50. What is the minimum length of telephone line required? Assume that no telephone line can be built between subdivisions 1 and 4.

FIGURE 50
Network for Problem 2



Group B

3 In this problem, we explain why the MST algorithm works. Define

S = minimum spanning tree

C_t = nodes connected after iteration t of MST algorithm has been completed

C'_t = nodes not connected after iteration t of MST algorithm has been completed

A_t = set of arcs in minimum spanning tree after t iterations of MST algorithm have been completed

TABLE 38

	Gary	Fort Wayne	Evansville	Terre Haute	South Bend
Gary	—	132	217	164	58
Fort Wayne	132	—	290	201	79
Evansville	217	290	—	113	303
Terre Haute	164	201	113	—	196
South Bend	58	79	303	196	—

Suppose the MST algorithm does not yield a minimum spanning tree. Then, for some t, it must be the case that all arcs in A_{t-1} are in S, but the arc chosen at iteration t (call it a_t) of the MST algorithm is not in S. Then S must contain some arc a'_t that leads from a node in C_{t-1} to a node in C'_{t-1}). Show that by replacing arc a'_t with arc a_t, we can obtain a shorter spanning tree than S. This contradiction proves that all arcs chosen by the MST algorithm must be in S. Thus, the MST algorithm does indeed find a minimum spanning tree.

4 **a** Three cities are at the vertices of an equilateral triangle of unit length. Flying Lion Airlines needs to supply connecting service between these three cities. What is the minimum length of the two routes needed to supply the connecting service?

b Now suppose Flying Lion Airlines adds a hub at the "center" of the equilateral triangle. Show that the length of the routes needed to connect the three cities has decreased by 13%. *(Note:* It has been shown that no matter how many "hubs" you add and no matter how many points must be connected, you can never save more than 13% of the total distance needed to "span" all the original points by adding hubs.)†

8.7 The Network Simplex Method‡

In this section, we describe how the simplex algorithm simplifies for MCNFPs. To simplify our presentation, we assume that for each arc, $L_{ij} = 0$. Then the information needed to describe an MCNFP of the form (8)–(9) may be summarized graphically as in Figure 51. We will denote the c_{ij} for each arc by the symbol \$, and the other number on each arc will represent the arc's upper bound (U_{ij}). The b_i for any node with nonzero outflow will be listed in parentheses. Thus, Figure 51 represents an MCNFP with $c_{12} = 5$, $c_{25} = 2$, $c_{13} = 4$, $c_{35} = 8$,

†Based on Peterson (1990).
‡This section covers topics that may be omitted with no loss of continuity.

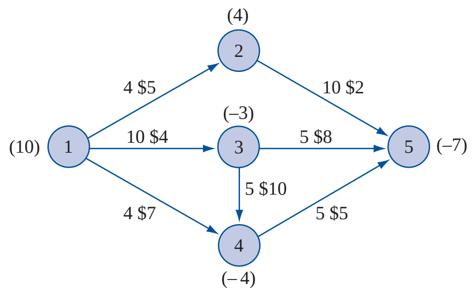


FIGURE 51 Graphical Representation of an MCNFP

$c_{14} = 7$, $c_{34} = 10$, $c_{45} = 5$, $b_1 = 10$, $b_2 = 4$, $b_3 = -3$, $b_4 = -4$, $b_5 = -7$, $U_{12} = 4$, $U_{25} = 10$, $U_{13} = 10$, $U_{35} = 5$, $U_{14} = 4$, $U_{34} = 5$, $U_{45} = 5$. For the network simplex to be used, we must have $\Sigma b_i = 0$; usually this can be ensured by adding a dummy node.

Recall that when we used the simplex method to solve a transportation problem, the following aspects of the simplex algorithm simplified: finding a basic feasible solution, computing the coefficient of a nonbasic variable in row 0, and pivoting. We now describe how these aspects of the simplex algorithm simplify when we are solving an MCNFP.

Basic Feasible Solutions for MCNFPs

How can we determine whether a feasible solution to an MCNFP is a bfs? Begin by observing that any bfs to an MCNFP will contain three types of variables:

1 Basic variables: In the absence of degeneracy, each basic variable x_{ij} will satisfy $L_{ij} < x_{ij} < U_{ij}$; with degeneracy, it is possible for a basic variable x_{ij} to equal arc (i, j)'s upper or lower bound.

2 Nonbasic variables x_{ij}: These equal arc (i, j)'s upper bound U_{ij}.

3 Nonbasic variables x_{ij}: These equal arc (i, j)'s lower bound L_{ij}.

Suppose we are solving an MCNFP with n nodes. In solving an MCNFP, we consider the n conservation-of-flow constraints and ignore the upper- and lower-bound constraints (for reasons that will soon become apparent). As in the transportation problem, any solution satisfying $n - 1$ of the conservation-of-flow constraints will automatically satisfy the last conservation-of-flow constraint, so we may drop one such constraint. This means that a bfs to an n-node MCNFP will have $n - 1$ basic variables. Suppose we choose a set of $n - 1$ variables (or arcs). How can we determine whether this set of $n - 1$ variables yields a basic feasible solution? A set of $n - 1$ variables will yield a bfs if and only if the arcs corresponding to the basic variables form a spanning tree for the network. For example, consider the MCNFP in Figure 52. In Figure 53, we give a bfs for this MCNFP. The basic variables are x_{13}, x_{35}, x_{25}, and x_{45}. The variables $x_{12} = 5$ and $x_{14} = 4$ are nonbasic vari-

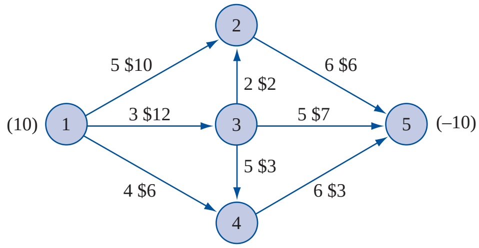


FIGURE 52 Example of an MCNFP

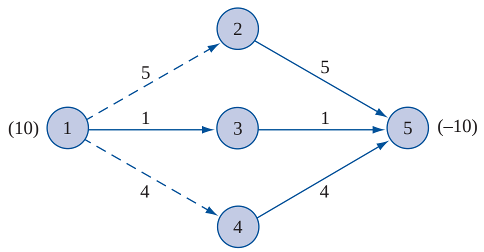


FIGURE 53
Example of a bfs for an MCNFP

ables at their upper bound. (Such variables will be indicated by dashed arcs.) Because the arcs (1, 3), (3, 5), (2, 5), and (4, 5) form a spanning tree (they connect all nodes of the graph and do not contain any cycles), we know that this is a bfs. As will soon become clear, a bfs for small problems can often be obtained by trial and error.

Computing Row 0 for Any bfs

For any given bfs, how do we determine the objective function coefficient for a nonbasic variable? Suppose we arbitrarily choose to drop the conservation-of-flow constraint for node 1. For a given bfs, let $c_{BV}B^{-1} = [y_2 \quad y_3 \quad \cdots \quad y_n]$. Each variable x_{ij} will have a $+1$ coefficient in the node i flow constraint and a -1 coefficient in the node j constraint. If we define $y_1 = 0$, then the coefficient of x_{ij} in row 0 of a given tableau may be written as $\bar{c}_{ij} = y_i - y_j - c_{ij}$. Each basic variable must have $\bar{c}_{ij} = 0$, so we can find $y_1, y_2, \ldots, y_n$ by solving the following system of linear equations:

$$y_1 = 0, \quad y_i - y_j = c_{ij} \quad \text{for each basic variable}$$

The $y_1, y_2, \ldots, y_n$ corresponding to a bfs are often called the **simplex multipliers** for the bfs.

How can we determine whether a bfs is optimal? For a bfs to be optimal, it must be possible to improve (decrease) the value of z by changing the value of a nonbasic variable. Note that $\bar{c}_{ij} \leq 0$ if and only if increasing x_{ij} cannot decrease z. Also note that $\bar{c}_{ij} \geq 0$ if and only if decreasing x_{ij} cannot decrease z. These observations can be used to show that a bfs is optimal if and only if the following conditions are met:

1 If a variable $x_{ij} = L_{ij}$, then an increase in x_{ij} cannot result in a decrease in z. Thus, if $x_{ij} = L_{ij}$ and the bfs is optimal, then $\bar{c}_{ij} \leq 0$ must hold.

2 If a variable $x_{ij} = U_{ij}$, then a decrease in x_{ij} cannot result in a decrease in z. Thus, if $x_{ij} = U_{ij}$ and the bfs is optimal, then $\bar{c}_{ij} \geq 0$ must hold.

If conditions 1 and 2 are not met, then z can be improved (barring degeneracy) by pivoting into the basis any nonbasic variable violating either condition. To illustrate, let's determine the objective function coefficient for each nonbasic variable in the simplex tableau corresponding to the bfs in Figure 53. To find y_1, y_2, y_3, y_4, and y_5, we solve the following set of equations:

$$y_1 = 0, \quad y_1 - y_3 = 12, \quad y_2 - y_5 = 6, \quad y_3 - y_5 = 7, \quad y_4 - y_5 = 3$$

The solutions to these equations are $y_1 = 0$, $y_2 = -13$, $y_3 = -12$, $y_4 = -16$, and $y_5 = -19$. We now "price out" each nonbasic variable and obtain

$\bar{c}_{12} = y_1 - y_2 - c_{12} = 0 - (-13) - 10 = 3$ (Satisfies optimality condition for nonbasic variable at upper bound)

$\bar{c}_{14} = y_1 - y_4 - c_{14} = 0 - (-16) - 6 = 10$ (Satisfies optimality condition for nonbasic variable at upper bound)

$$\bar{c}_{32} = y_3 - y_2 - c_{32} = -12 - (-13) - 2 = -1 \quad \text{(Satisfies optimality condition for nonbasic variable at lower bound)}$$

$$\bar{c}_{34} = y_3 - y_4 - c_{34} = -12 - (-16) - 3 = 1 \quad \text{(Violates optimality condition for nonbasic variable at lower bound)}$$

Because $\bar{c}_{34} = 1 > 0$, each unit by which we increase x_{34} (x_{34} is at its lower bound, so it's okay to increase it) will decrease z by one unit. Thus, we can improve z by entering x_{34} into the basis. Note that if a nonbasic variable x_{ij} at its upper bound had $\bar{c}_{ij} < 0$, then we could decrease z by entering x_{ij} into the basis and decreasing x_{ij}. We now show that when solving an MCNFP, the pivot step may be performed almost by inspection.

Pivoting in the Network Simplex

As we have just shown, for the bfs in Figure 53, we want to enter x_{34} into the basis. To do this, note that if we add the arc (3, 4) to the set of arcs corresponding to the current set of basic variables, a cycle (or loop) will be formed. To enter x_{34} into the basis, note that $x_{34} = 0$ is at its lower bound, we want to increase x_{34}. Suppose we try to increase x_{34} by θ. The values of all variables after x_{34} is entered into the basis may be found by invoking the conservation-of-flow constraints. In Figure 54, we find that arc (3, 4), (4, 5), and (3, 5) form a cycle. After the pivot, all variables corresponding to arcs not in the cycle will remain unchanged, but when we set $x_{34} = \theta$, the values of the variables corresponding to arcs in the cycle will change. Setting $x_{34} = \theta$ increases the flow into node 4 by θ, so the flow out of node 4 must increase by θ. This requires $x_{45} = 4 + \theta$. Because the flow into node 5 has now increased by θ, conservation of flow requires that $x_{35} = 1 - \theta$. The pivot leaves all other variables unchanged. To find the new values of the variables, observe that we want to increase x_{34} by as much as possible. We can increase x_{34} to the point where a basic variable first attains its upper or lower bound. Thus, arc (3, 4) implies that $\theta \le 5$; arc (3, 5) requires $1 - \theta \ge 0$ or $\theta \le 1$; arc (4, 5) requires $4 + \theta \le 6$ or $\theta \le 2$. So the best we can do is set $\theta = 1$. The basic variable that first hits its upper or lower bound as θ is increased is chosen to exit the basis (in case of a tie, we can choose the exiting variable arbitrarily). Now x_{35} exits the basis, and the new bfs is shown in Figure 55. The spanning tree corresponding to the current set of basic variables is (1, 3), (3, 4), (4, 5), and (2, 5). We now compute the coefficient of each nonbasic variable in row 0. To begin, we solve the following set of equations:

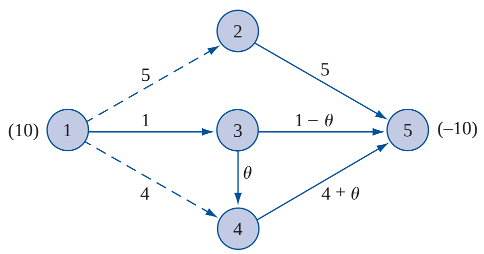
$$y_1 = 0, \qquad y_1 - y_3 = 12, \qquad y_3 - y_4 = 3, \qquad y_2 - y_5 = 6, \qquad y_4 - y_5 = 3$$

This yields $y_1 = 0$, $y_2 = -12$, $y_3 = -12$, $y_4 = -15$, and $y_5 = -18$.

The nonbasic variables that currently equal their upper bounds will have row 0 coefficients of

$$\bar{c}_{12} = 0 - (-12) - 10 = 2 \qquad \text{and} \qquad \bar{c}_{14} = 0 - (-15) - 6 = 9$$

FIGURE 54
Cycle (3, 4), (4, 5), (3, 5) Helps Us Pivot in x_{34}



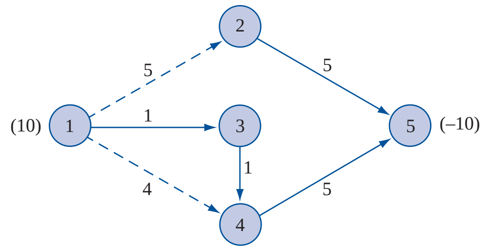


FIGURE 55
New bfs ($\theta = 1$) After x_{34} Enters and x_{35} Exits

The nonbasic variables that currently equal their lower bounds will have row 0 coefficients of

$$\bar{c}_{32} = -12 - (-12) - 2 = -2 \qquad \text{and} \qquad \bar{c}_{35} = -12 - (-18) - 7 = -1$$

Because each nonbasic variable at its upper bound has $\bar{c}_{ij} \geq 0$, and each nonbasic variable at its lower bound has $\bar{c}_{ij} \leq 0$, the current bfs is optimal. Thus, the optimal solution to the MCNFP in Figure 52 is

Upper bounded variables: $x_{12} = 5, \quad x_{14} = 4$
Lower bounded variables: $x_{32} = x_{35} = 0$
Basic variables: $x_{13} = 1, x_{34} = 1, x_{25} = 5, x_{45} = 5$

Summary of the Network Simplex Method

Step 1 Determine a starting bfs. The $n - 1$ basic variables will correspond to a spanning tree. Indicate nonbasic variables at their upper bound by dashed arcs.

Step 2 Compute $y_1, y_2, \ldots y_n$ (often called the simplex multipliers) by solving $y_1 = 0$, $y_i - y_j = c_{ij}$ for all basic variables x_{ij}. For all nonbasic variables, determine the row 0 coefficient $\bar{c}_{ij}$ from $\bar{c}_{ij} = y_i - y_j - c_{ij}$. The current bfs is optimal if $\bar{c}_{ij} \leq 0$ for all $x_{ij} = L_{ij}$ and $\bar{c}_{ij} \geq 0$ for all $x_{ij} = U_{ij}$. If the bfs is not optimal, choose the nonbasic variable that most violates the optimality conditions as the entering basic variable.

Step 3 Identify the cycle (there will be exactly one!) created by adding the arc corresponding to the entering variable to the current spanning tree of the current bfs. Use conservation of flow to determine the new values of the variables in the cycle. The variable that exits the basis will be the variable that first hits its upper or lower bound as the value of the entering basic variable is changed.

Step 4 Find the new bfs by changing the flows of the arcs in the cycle found in step 3. Now go to step 2.

Example 9 illustrates the network simplex.

EXAMPLE 9 Network Simplex Solution to MCNFP

Use the network simplex to solve the MCNFP in Figure 56.

Solution A bfs requires that we find a spanning tree (three arcs that connect nodes 1, 2, 3, and 4 and do not form a cycle). Any arcs not in the spanning tree may be set equal to their upper or lower bound. By trial and error, we find the bfs in Figure 57 involving the spanning tree (1, 2), (1, 3), and (2, 4).

To find y_1, y_2, y_3, and y_4 we solve

$$y_1 = 0, \qquad y_1 - y_2 = 4, \qquad y_2 - y_4 = 3, \qquad y_1 - y_3 = 3$$

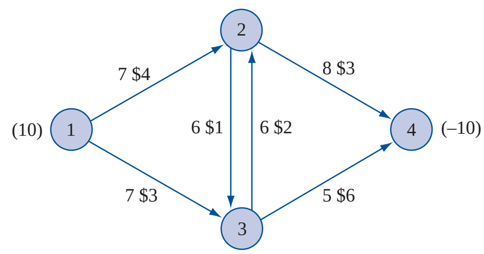


FIGURE 56
Example of Network Simplex

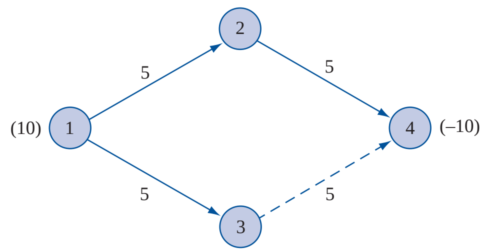


FIGURE 57
bfs for Example 9

This yields $y_1 = 0$, $y_2 = -4$, $y_3 = -3$, and $y_4 = -7$. The row 0 coefficients for each nonbasic variable are

$$\bar{c}_{34} = -3 - (-7) - 6 = -2 \quad \text{(Violates optimality condition)}$$
$$\bar{c}_{23} = -4 - (-3) - 1 = -2 \quad \text{(Satisfies optimality condition)}$$
$$\bar{c}_{32} = -3 - (-4) - 2 = -1 \quad \text{(Satisfies optimality condition)}$$

Thus, x_{34} enters the basis. We set $x_{34} = 5 - \theta$ and obtain the cycle in Figure 58. From arc (1, 2), we find $5 + \theta \leq 7$ or $\theta \leq 2$. From arc (1, 3), we find $5 - \theta \geq 0$ or $\theta \leq 5$. From arc (2, 4), we find $5+ \theta \leq 8$ or $\theta \leq 3$. From arc (3, 4), we find $5 - \theta \geq 0$ or $\theta \leq 5$. Thus, we can set $\theta = 2$. Now x_{12} exits the basis at its upper bound, and x_{34} enters, yielding the bfs in Figure 59.

The new bfs is associated with the spanning tree (1, 3), (2, 4), and (3, 4). Solving for the new values of the simplex multipliers, we obtain

$$y_1 = 0, \quad y_1 - y_3 = 3, \quad y_3 - y_4 = 6, \quad y_2 - y_4 = 3$$

This yields $y_1 = 0$, $y_2 = -6$, $y_3 = -3$, $y_4 = -9$. The coefficient of each nonbasic variable in row 0 is given by

$$\bar{c}_{12} = 0 - (-6) - 4 = 2 \quad \text{(Satisfies optimality condition)}$$
$$\bar{c}_{23} = -6 - (-3) - 1 = -4 \quad \text{(Satisfies optimality condition)}$$
$$\bar{c}_{32} = -3 - (-6) - 2 = 1 \quad \text{(Violates optimality condition)}$$

Now x_{32} enters the basis, yielding the cycle in Figure 60. From arc (2, 4), we find $7 + \theta \leq 8$ or $\theta \leq 1$); from arc (3, 4), we find $3 - \theta \geq 0$ or $\theta \leq 3$. From arc (3, 2), we find $\theta \leq 6$. So we now set $\theta = 1$ and have x_{24} exit from the basis at its upper bound. The new bfs is given in Figure 61.

The current set of basic values corresponds to the spanning tree (1, 3), (3, 2), and (3, 4). The new values of the simplex multipliers are found by solving

$$y_1 = 0, \quad y_1 - y_3 = 3, \quad y_3 - y_2 = 2, \quad y_3 - y_4 = 6$$

which yields $y_1 = 0$, $y_2 = -5$, $y_3 = -3$, $y_4 = -9$. The coefficient of each nonbasic variable in row 0 is now

1091.ch08 5/13/03 1:23 PM Page 465

FIGURE 58
Cycle Created When x_{34} Enters the Basis

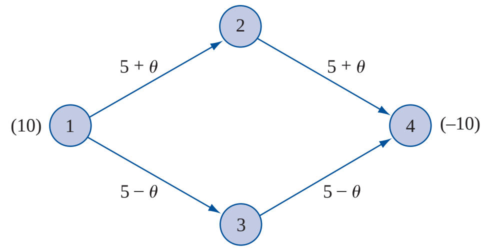


FIGURE 59
bfs After x_{12} Exits and x_{34} Enters

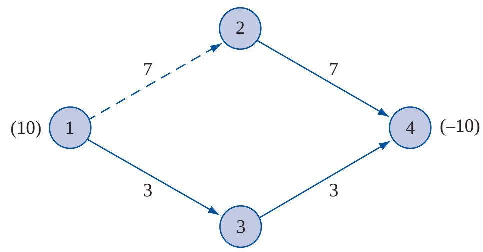


FIGURE 60
Cycle Created When x_{32} Enters Basis

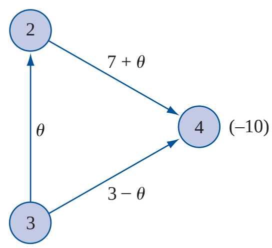
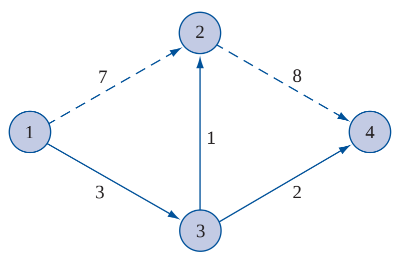


FIGURE 61
New bfs When x_{32} Enters and x_{24} Exits



$$\bar{c}_{23} = -5 - (-3) - 1 = -3 \qquad \text{(Satisfies optimality condition)}$$
$$\bar{c}_{12} = 0 - (-5) - 4 = 1 \qquad \text{(Satisfies optimality condition)}$$
$$\bar{c}_{24} = -5 - (-9) - 3 = 1 \qquad \text{(Satisfies optimality condition)}$$

Thus, the current bfs is optimal. The optimal solution to the MCNFP is

Basic variables: $x_{13} = 3,\quad x_{32} = 1,\quad x_{34} = 2$
Nonbasic variables at their upper bound: $x_{12} = 7,\quad x_{24} = 8$
Nonbasic variable at lower bound: $x_{23} = 0$

The optimal z-value is obtained from

$$z = 7(4) + 3(3) + 1(2) + 8(3) + 2(6) = \$75$$

PROBLEMS

Group A

1 Consider the problem of finding the shortest path from node 1 to node 6 in Figure 2.

a Formulate this problem as an MCNFP.

b Find a bfs in which x_{12}, x_{24}, and x_{46} are positive. (*Hint:* A degenerate bfs will be obtained.)

c Use the network simplex to find the shortest path from node 1 to node 6.

2 For the MCNFP in Figure 62, find a bfs.

3 Find the optimal solution to the MCNFP in Figure 63 using the bfs in Figure 64 as a starting basis.

FIGURE 62

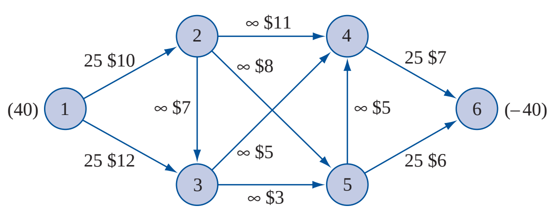


FIGURE 63

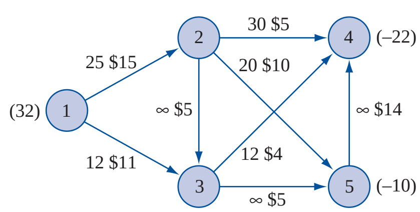
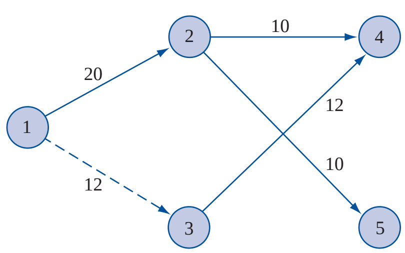


FIGURE 64



4 Find a bfs for the network in Figure 65.

5 Find the optimal solution to the MCNFP in Figure 66 using the bfs in Figure 67 as a starting basis.

FIGURE 65

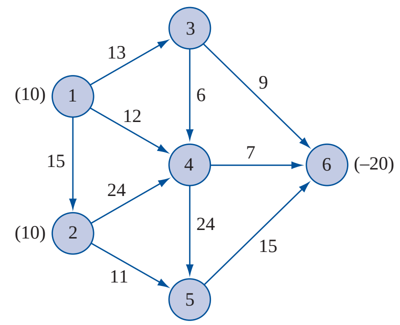


FIGURE 66

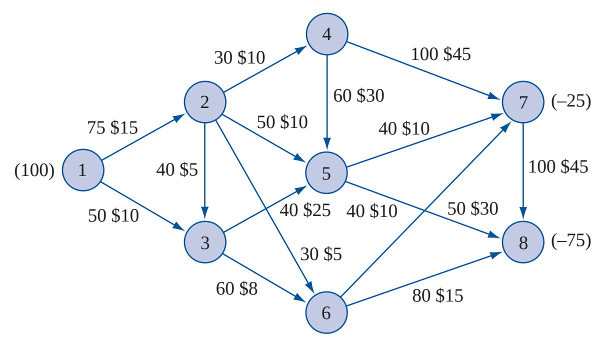
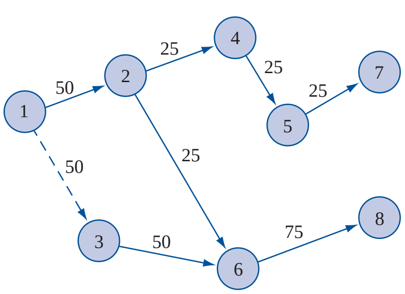


FIGURE 67



SUMMARY

Shortest-Path Problems

Suppose we want to find the shortest path from node 1 to node j in a network in which all arcs have nonnegative lengths.

Dijkstra's Algorithm

1 Label node 1 with a permanent label of 0. Then label each arc connected to node 1 by a single arc with a "temporary" label equal to the length of the arc joining node 1 and node i. Remaining nodes will have a temporary label of ∞. Choose the node with the smallest temporary label and make this label permanent.

2 Suppose that node i is the $(k + 1)$th node to be given a permanent label. For each node j that now has a temporary label and is connected to node i by an arc, replace node j's temporary label with min {node j's current temporary label, (node i's permanent label) + length of arc (i, j)}. Make the smallest temporary label a permanent label. Continue this process until all nodes have permanent labels. To find the shortest path from node 1 to node j, work backward from node j by finding nodes having labels differing by exactly the length of the connecting arc. If the shortest path from node 1 to node j is desired, stop the labeling process as soon as node j receives a permanent label.

The Shortest-Path Problem as a Transshipment Problem

To find the shortest path from node 1 to node j, try to minimize the cost of sending one unit from node 1 to node j (with all other nodes in the network being transshipment points), where the cost of sending one unit from node k to node k' is the length of arc (k, k') if such an arc exists and is M (a large positive number) if such an arc does not exist. As in Section 7.6, the cost of shipping one unit from a node to itself is zero.

Maximum-Flow Problems

We can find the maximum flow from source to sink in a network by linear programming or by the Ford–Fulkerson method.

Finding Maximum Flow by Linear Programming

Let

$$x_0 = \text{flow through artificial arc going from sink to source}$$

Then to find the maximum flow from source to sink, maximize x_0 subject to the following two sets of constraints:

1 The flow through each arc must be nonnegative and cannot exceed the arc capacity.

2 Flow into node i = flow out of node i (Conservation of flow)

Finding Maximum Flow by the Ford–Fulkerson Method

Let

$$I = \text{set of arcs in which flow may be increased}$$
$$R = \text{set of arcs in which flow may be reduced}$$

Step 1 Find a feasible flow (setting each arc's flow to zero will do).

Step 2 Using the following procedure, try to find a chain of labeled arcs and nodes that can be used to label the sink. Label the source. Then label vertices and arcs (except for arc a_0) according to the following rules: (1) If vertex x is labeled, then vertex y is unlabeled and arc (x, y) is a member of I; then label vertex y and arc (x, y). Arc (x, y) is called a **forward arc.** (2) If vertex y is unlabeled, then vertex x is labeled and arc (y, x) is a member of R; then label vertex y and arc (y, x). Arc (y, x) is called a **backward arc.**

If the sink cannot be labeled, the current feasible flow is a maximum flow; if the sink is labeled, go on to step 3.

Step 3 If the chain used to label the sink consists entirely of forward arcs, the flow through each of the forward arcs in the chain may be increased, thereby increasing the flow from source to sink. If the chain used to label the sink consists of both forward and backward arcs, increase the flow in each forward arc in the chain and decrease the flow in each backward arc in the chain. Again, this will increase the flow from source to sink. Return to step 2.

Critical Path Method

Assuming the duration of each activity is known, the critical path method (CPM) may be used to find the duration of a project.

Rules for Constructing an AOA Project Diagram

1 Node 1 represents the start of the project. An arc should lead from node 1 to represent each activity that has no predecessors.

2 A node (called the finish node) representing the completion of the project should be included in the network.

3 Number the nodes in the network so that the node representing the completion of an activity always has a larger number than the node representing the beginning of an activity (there may be more than one numbering scheme that satisfies rule 3).

4 An activity should not be represented by more than one arc in the network.

5 Two nodes can be connected by at most one arc.

To avoid violating rules 4 and 5, it is sometimes necessary to utilize a **dummy activity** that takes zero time.

Computation of Early Event Time

The early event time for node i, denoted $ET(i)$, is the earliest time at which the event corresponding to node i can occur. We compute $ET(i)$ as follows:

Step 1 Find each prior event to node i that is connected by an arc to node i. These events are the **immediate predecessors** of node i.

Step 2 To the ET for each immediate predecessor of node i, add the duration of the activity connecting the immediate predecessor to node i.

Step 3 $ET(i)$ equals the maximum of the sums computed in step 2.

Computation of Late Event Time

The late event time for node i, denoted $LT(i)$, is the latest time at which the event corresponding to node i can occur without delaying the completion of the project. We compute $LT(i)$ as follows:

Step 1 Find each node that occurs after node i and is connected to node i by an arc. These events are the **immediate successors** of node i.

Step 2 From the LT for each immediate successor to node i, subtract the duration of the activity joining the successor to node i.

Step 3 $LT(i)$ is the smallest of the differences determined in step 2.

Total Float

For an arbitrary arc representing activity (i, j), the total float (denoted $TF(i, j)$ of the activity represented by (i, j) is the amount by which the starting time of activity (i, j) could be delayed beyond its earliest possible starting time without delaying the completion of the project (assuming no other activities are delayed):

$$TF(i, j) = LT(j) - ET(i) - t_{ij} \quad [t_{ij} = \text{duration of activity represented by arc } (i, j)]$$

Any activity with a total float of zero is a **critical activity.** A path from node 1 to the finish node that consists entirely of critical activities is called a **critical path.** Any critical path (there may be more than one in a project network) is the longest path in the network from the start node (node 1) to the finish node. If the start of a critical activity is delayed, or if the duration of a critical activity is longer than expected, then the completion of the project will be delayed.

Free Float

The free float of the activity corresponding to arc (i, j), denoted by $FF(i, j)$, is the amount by which the starting time of the activity corresponding to arc (i, j) (or the duration of the activity) can be delayed without delaying the start of any later activity beyond its earliest possible starting time:

$$FF(i, j) = ET(j) - ET(i) - t_{ij}$$

Linear programming can be used to find a critical path and the duration of the project. Let

$$x_j = \text{time at which node } j \text{ in project network occurs}$$
$$F = \text{node representing finish or completion of the project}$$

To find a critical path, minimize $z = x_F - x_1$ subject to

$$x_j \geq x_i + t_{ij} \quad \text{or} \quad x_j - x_i \geq t_{ij} \quad \text{for each arc}$$
$$x_j \text{ urs}$$

The optimal objective function value is the length of any critical path (or time to project completion). To find a critical path, simply find a path from node 1 to node F for which each arc in the path is represented by an arc (i, j) whose constraint $(x_j - x_i \geq t_{ij})$ has a dual price of -1.

Linear programming can also be used to determine the minimum-cost method of reducing the duration of activities (crashing) to meet a project completion deadline.

PERT

If the durations of the project's activities are not known with certainty, then PERT may be used to estimate the probability that the project will be completed in a specified amount of time. PERT requires that for each activity the following three numbers be specified:

a = estimate of the activity's duration under the most favorable conditions

b = estimate of the activity's duration under the least favorable conditions

m = most likely value for the activity's duration

If the estimates a, b, and m refer to the activity represented by arc (i, j), then $\mathbf{T}_{ij}$ is the random variable representing the duration of the activity represented by arc (i, j). $\mathbf{T}_{ij}$ has (approximately) the following properties:

$$E(\mathbf{T}_{ij}) = \frac{a + 4m + b}{6}$$

$$\text{var}\mathbf{T}_{ij} = \frac{(b - a)^2}{36}$$

Then

$$\sum_{(i,\, j)\in\text{path}} E(\mathbf{T}_{ij}) = \text{expected duration of activities on any path}$$

$$\sum_{(i,\, j)\in\text{path}} \text{var}\mathbf{T}_{ij} = \text{variance of duration of activities on any path}$$

Assuming (sometimes incorrectly) that the critical path found by CPM is the critical path, and assuming that the duration of the critical path is normally distributed, the preceding equations may be used to estimate the probability that the project will be completed within any specified length of time.

Minimum-Cost Network Flow Problems

The transportation, assignment, transshipment, shortest-path, maximum-flow, and critical path problems are all special cases of the minimum-cost network flow problem (MCNFP).

x_{ij} = number of units of flow sent from node i to node j through arc (i, j)

b_i = net supply (outflow − inflow) at node i

c_{ij} = cost of transporting one unit of flow from node i to node j via arc (i, j)

L_{ij} = lower bound on flow through arc (i, j) (if there is no lower bound, let $L_{ij} = 0$)

U_{ij} = upper bound on flow through arc (i, j) (if there is no upper bound, let $U_{ij} = \infty$)

Then an MCNFP may be written as

$$\min \sum_{\text{all arcs}} c_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{ij} - \sum_k x_{ki} = b_i \qquad \text{(for each node } i \text{ in the network)}$$

$$L_{ij} \le x_{ij} \le U_{ij} \qquad \text{(for each arc in the network)}$$

The first set of constraints are the **flow balance equations,** and the second set of constraints express limitations on arc capacities.

Any MCNFP may be solved by a computer code using the **network simplex;** the user need only input the nodes and arcs in the network, the c_{ij}'s and arc capacity for each arc, and the b_i's for each node. Formulation of a problem as an MCNFP may require adding a dummy point to the problem.

Minimum Spanning Tree Problems

The following method (MST algorithm) may be used to find a minimum spanning tree for a network:

Step 1 Begin at any node i, and join node i to the node in the network (node j) that is closest to node i. The two nodes i and j now form a connected set of nodes $C = \{i, j\}$ and arc (i, j) will be in the minimum spanning tree. The remaining nodes in the network (C') are the unconnected set of nodes.

Step 2 Choose a member of $C'(n)$ that is closest to some node in C. Let m represent the node in C that is closest to n. Then the arc (m, n) will be in the minimum spanning tree. Update C and C'. Because n is now connected to $\{i, j\}$, C now equals $\{i, j, n\}$, and we must eliminate node n from C'.

Step 3 Repeat this process until a minimum spanning tree is found. Ties for closest node and arc may be broken arbitrarily.

Network Simplex Method

Step 1 Determine a starting bfs. The $n - 1$ basic variables will correspond to a spanning tree. Indicate nonbasic variables at their upper bound by dashed arcs.

Step 2 Compute $y_1, y_2, \ldots y_n$ (often called the *simplex multipliers*) by solving $y_1 = 0$, $y_i - y_j = c_{ij}$ for all basic variables x_{ij}. For all nonbasic variables, determine the row 0 coefficient $\bar{c}_{ij}$ from $\bar{c}_{ij} = y_i - y_j - c_{ij}$. The current bfs is optimal if $\bar{c}_{ij} \le 0$ for all $x_{ij} = L_{ij}$ and $\bar{c}_{ij} \ge 0$ for all $x_{ij} = U_{ij}$. If the bfs is not optimal, then choose the nonbasic variable that most violates the optimality conditions as the entering basic variable.

Step 3 Identify the cycle (there will be exactly one!) created by adding the arc corresponding to the entering variable to the current spanning tree of the current bfs. Use conservation of flow to determine the new values of the variables in the cycle. The variable that first hits its upper or lower bound as the value of the entering basic variable is changed exits the basis.

Step 4 Find the new bfs by changing the flows of the arcs in the cycle found in step 3. Go to step 2.

REVIEW PROBLEMS

Group A

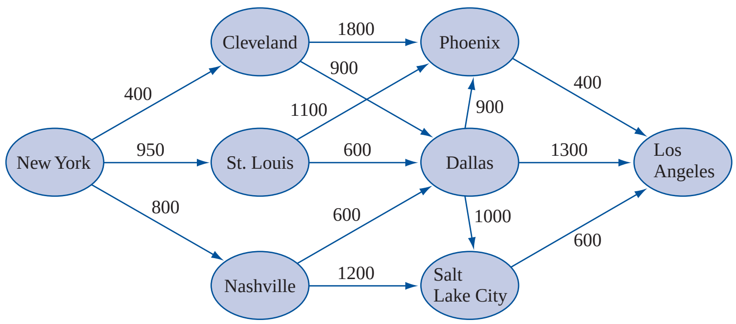
1 A truck must travel from New York to Los Angeles. As shown in Figure 68, a variety of routes are available. The number associated with each arc is the number of gallons of fuel required by the truck to traverse the arc.

a Use Dijkstra's algorithm to find the route from New York to Los Angeles that uses the minimum amount of gas.

b Formulate a balanced transportation problem that could be used to find the route from New York to Los Angeles that uses the minimum amount of gas.

c Formulate as an MCNFP the problem of finding the New York to Los Angeles route that uses the minimum amount of gas.

FIGURE 68
Network for Problem 1



2 Telephone calls from New York to Los Angeles are transported as follows: The call is sent first to either Chicago or Memphis, then routed through either Denver or Dallas, and finally sent to Los Angeles. The number of phone lines joining each pair of cities is shown in Table 39.

a Formulate an LP that can be used to determine the maximum number of calls that can be sent from New York to Los Angeles at any given time.

b Use the Ford–Fulkerson method to determine the maximum number of calls that can be sent from New York to Los Angeles at any given time.

TABLE 39

Cities	No. of Telephone Lines
N.Y.–Chicago	500
N.Y.–Memphis	400
Chicago–Denver	300
Chicago–Dallas	250
Memphis–Denver	200
Memphis–Dallas	150
Denver–L.A.	400
Dallas–L.A.	350

3 Before a new product can be introduced, the activities in Table 40 must be completed (all times are in weeks).

a Draw the project diagram.

b Determine all critical paths and critical activities.

c Determine the total float and free float for each activity.

d Set up an LP that can be used to determine the critical path.

e Formulate an MCNFP that can be used to find the critical path.

f It is now 12 weeks before Christmas. What is the probability that the product will be in the stores before Christmas?

g The duration of each activity can be reduced by up to 2 weeks at the following cost per week: A, \$80; B, \$60; C, \$30; D, \$60; E, \$40; F, \$30; G, \$20. Assuming that the duration of each activity is known with certainty, formulate an LP that will minimize the cost of getting the product into the stores by Christmas.

4 During the next three months, Shoemakers, Inc. must meet (on time) the following demands for shoes: month 1, 1,000 pairs; month 2, 1,500 pairs; month 3, 1,800 pairs. It takes 1 hour of labor to produce a pair of shoes. During each of the next three months, the following number of regular-time labor hours are available: month 1, 1,000 hours; month 2, 1,200 hours; month 3, 1,200 hours. Each month, the company can require workers to put in up to 400 hours of overtime. Workers

TABLE 40

Activity	Description	Predecessors	Duration	*a*	*b*	*m*
A	Design the product	—	6	2	10	6
B	Survey the market	—	5	4	6	5
C	Place orders for raw materials	A	3	2	4	3
D	Receive raw materials	C	2	1	3	2
E	Build prototype of product	A, D	3	1	5	3
F	Develop ad campaign	B	2	3	5	4
G	Set up plan for mass production	E	4	2	6	4
H	Deliver product to stores	G, F	2	0	4	2

are paid only for the hours they work, and a worker receives \$4 per hour for regular-time work and \$6 per hour for overtime work. At the end of each month, a holding cost of \$1.50 per pair of shoes is incurred. Formulate an MCNFP that can be used to minimize the total cost incurred in meeting the demands of the next three months. A formulation requires drawing the appropriate network and determining the c_{ij}'s, b_i's, and arc capacities. How would you modify your answer if demand could be backlogged (all demand must still be met by the end of month 3) at a cost of \$20/pair/month?

5 Find a minimum spanning tree for the network in Figure 68.

6 A company produces a product at two plants, 1 and 2. The unit production cost and production capacity during each period are given in Table 41. The product is instantaneously shipped to the company's only customer according to the unit shipping costs given in Table 42. If a unit is produced and shipped during period 1, it can still be used to meet a period 2 demand, but a holding cost of \$13 per unit in inventory is assessed. At the end of period 1, at most six units may be held in inventory. Demands are as follows: period 1, 9; period 2, 11. Formulate an MCNFP that can be used to minimize the cost of meeting all demands on time. Draw the network and determine the net outflow at each node, the arc capacities, and shipping costs.

7 A project is considered completed when activities A–F have all been completed. The duration and predecessors of each activity are given in Table 43. The LINDO output in Figure 69 can be used to determine the critical path for this project.

a Use the LINDO output to draw the project network. Indicate the activity represented by each arc.

b Determine a critical path in the network. What is the earliest the project can be completed?

8[†] State University has three professors who each teach four courses per year. Each year, four sections of marketing, finance, and production must be offered. At least one section of each class must be offered during each semester (fall and spring). Each professor's time preference and preference for teaching various courses are given in Table 44.

TABLE 41

	Unit Production Cost (\$)	Capacity
Plant 1 (period 1)	33	7
Plant 1 (period 2)	43	4
Plant 2 (period 1)	30	9
Plant 2 (period 2)	41	9

TABLE 42

	Period 1	Period 2
Plant 1 to customer	\$51	\$60
Plant 2 to customer	\$42	\$71

[†]Based on Mulvey (1979).

FIGURE 69

```
MIN     X6 - X1
SUBJECT TO
       2) - X1 + X3 >=   3
       3)   X4 - X2 >=   1
       4) - X3 + X4 >=   0
       5) - X4 + X5 >=   7
       6) - X3 + X5 >=   5
       7)   X6 - X5 >=   5
       8)   X3 - X2 >=   0
       9) - X1 + X2 >=   2
END

   LP OPTIMUM FOUND AT STEP   3

        OBJECTIVE FUNCTION VALUE

1)         15.0000000

VARIABLE        VALUE          REDUCED COST
       X6       15.000000          0.000000
       X1        0.000000          0.000000
       X3        3.000000          0.000000
       X4        3.000000          0.000000
       X2        2.000000          0.000000
       X5       10.000000          0.000000

ROW         SLACK OR SURPLUS     DUAL PRICES
       2)        0.000000         -1.000000
       3)        0.000000          0.000000
       4)        0.000000         -1.000000
       5)        0.000000         -1.000000
       6)        2.000000          0.000000
       7)        0.000000         -1.000000
       8)        1.000000          0.000000
       9)        0.000000          0.000000

NO. ITERATIONS=      3
```

TABLE 43

Activity	Duration	Immediate Predecessors
A	2	—
B	3	—
C	1	A
D	5	A, B
E	7	B, C
F	5	D, E

The total satisfaction a professor earns teaching a class is the sum of the semester satisfaction and the course satisfaction. Thus, professor 1 derives a satisfaction of 3 + 6 = 9 from teaching marketing during the fall semester. Formulate an MCNFP that can be used to assign professors to courses so as to maximize the total satisfaction of the three professors.

Group B

9[†] During the next two months, Machineco must meet (on time) the demands for three types of products shown in Table 45. Two machines are available to produce these

[†]This problem is based on Brown, Geoffrion, and Bradley (1981).

TABLE 44

	Professor 1	Professor 2	Professor 3
Fall Preference	3	5	4
Spring Preference	4	3	4
Marketing	6	4	5
Finance	5	6	4
Production	4	5	6

TABLE 45

Month	Product 1	Product 2	Product 3
1	50 units	70 units	80 units
2	60 units	90 units	120 units

products. Machine 1 can only produce products 1 and 2, and machine 2 can only produce products 2 and 3. Each machine can be used for up to 40 hours per month. Table 46 shows the time required to produce one unit of each product (independent of the type of machine); the cost of producing one unit of each product on each type of machine; and the cost of holding one unit of each product in inventory for one month. Formulate an MCNFP that could be used to minimize the total cost of meeting all demands on time.

TABLE 46

Product	Production Time (minutes)	Production Cost ($) Machine 1	Production Cost ($) Machine 2	Holding Cost ($)
1	30	40	—	15
2	20	45	60	10
3	15	—	55	5

REFERENCES

Brown, G., A. Geoffrion, and G. Bradley. "Production and Sales Planning with Limited Shared Tooling at the Key Operation," *Management Science* 27(1981):247–259.

Glover, F., et al. "The Passenger-Mix Problem in the Scheduled Airlines," *Interfaces* 12(1982):73–80.

Mulvey, M. "Strategies in Modeling: A Personnel Example," *Interfaces* 9(no. 3, 1979):66–75.

Peterson, I. "Proven Path for Limiting Shortest Shortcut," *Science News* December 22, 1990: 389.

Ravidran, A. "On Compact Book Storage in Libraries," *Opsearch* 8(1971).

The following three texts contain an overview of networks at an elementary level:

Chachra, V., P. Ghare, and J. Moore. *Applications of Graph Theory Algorithms.* New York: North-Holland, 1979.

Mandl, C. *Applied Network Optimization.* Orlando, Fla.: Academic Press, 1979.

Phillips, D., and A. Diaz. *Fundamentals of Network Analysis.* Englewood Cliffs, N.J.: Prentice Hall, 1981.

The two best comprehensive references on network models are:

Ahuja, R., Magnanti, T., and Orlin, J. *Network Flows: Theory Algorithms and Applications.* Englewood-Cliffs, N.J.: Prentice-Hall, 1993.

Bersetkas, D. *Linear Network Optimization: Algorithms and Codes.* Cambridge, Mass.: MIT Press, 1991.

Detailed discussion of methods for solving shortest path problems can be found in the following three texts:

Denardo, E. *Dynamic Programming: Theory and Applications.* Englewood Cliffs, N.J.: Prentice Hall, 1982.

Evans, T., and E. Minieka. *Optimization Algorithms for Networks and Graphs.* New York: Dekker, 1992. Also discusses minimum spanning tree algorithms.

Hu, T. *Combinatorial Algorithms.* Reading, Mass.: Addison-Wesley, 1982. Also discusses minimum spanning tree algorithms.

Evans and Minieka (1992) and Hu (1982) discuss the maximum-flow problem in detail, as do the following three texts:

Ford, L., and D. Fulkerson. *Flows in Networks.* Princeton, N.J.: Princeton University Press, 1962.

Jensen, P., and W. Barnes. *Network Flow Programming.* New York: Wiley, 1980.

Lawler, E. *Combinatorial Optimization: Networks and Matroids.* Chicago: Holt, Rinehart & Winston, 1976.

Excellent discussions of CPM and PERT are contained in:

Hax, A., and D. Candea. *Production and Inventory Management.* Englewood Cliffs, N.J.: Prentice Hall, 1984.

Wiest, J., and F. Levy. *A Management Guide to PERT/CPM,* 2d ed. Englewood Cliffs, N.J.: Prentice Hall, 1977.

Jensen and Barnes (1980) and the following references each contain a detailed discussion of the network simplex method used to solve an MCNFP.

Chvàtal, V. *Linear Programming.* San Francisco: Freeman, 1983.

Shapiro, J. *Mathematical Programming: Structures and Algorithms.* New York: Wiley, 1979.

Wu, N., and R. Coppins. *Linear Programming and Extensions.* New York: McGraw-Hill, 1981.

An excellent discussion of applications of MCNFPs is contained in the following:

Glover, F., D. Klingman, and N. Phillips. *Network Models and Their Applications in Practice.* New York: Wiley, 1992.